h e g

Haute école de gestion
Genève

Utilisation de Snort dans une PME

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

par :

Ilir Kadriu

Conseiller au travail de Bachelor :

Ciarán Bryce

Genève, le 07 Juin 2019

Haute École de Gestion de Genève (HEG-GE)

Filière Informatique de Gestion

Hes·so GENÈVE
Haute Ecole Spécialisée
de Suisse occidentale

Déclaration

Ce travail de Bachelor est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre Bachelor of Science en Informatique de gestion.

L'étudiant a envoyé ce document par email à l'adresse remise par son conseiller au travail de Bachelor pour analyse par le logiciel de détection de plagiat URKUND, selon la procédure détaillée à l'URL suivante : http://www.urkund.fr/student_gorsahar.asp.

L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J'atteste avoir réalisé seul< e > le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 07 juin 2019

Ilir KADRIU

I. Kadriu

Remerciements

Je tiens à remercier mon directeur de mémoire, Monsieur Ciarán Bryce pour son aide précieuse et le temps consacré dans la recherche et la mise en œuvre de mon travail. Je tenais aussi à remercier les différents professeurs et assistant(e) qui m'ont apporté de l'aide dans la compréhension de mon sujet : Monsieur Jean-Luc Sarrade, Monsieur Gérard Ineichen, Monsieur Djavan Sergent et Madame Sara Gloor.

Par ailleurs, je souhaite remercier mes parents et mon entourage, notamment Monsieur Mathieu Legrand, qui m'ont soutenu et aidé lors de ces années d'études à la Haute école de gestion.

Résumé

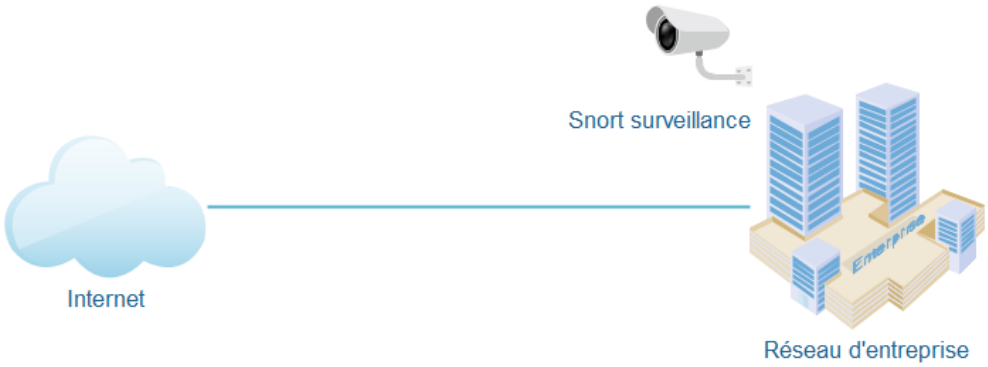
Les technologies informatiques se développent, les entreprises numérisent de plus en plus leurs données, et les réseaux d'entreprises s'agrandissent. Ce phénomène entraîne, d'une part, l'avancée des attaques cybernétiques, et d'autre part, la recherche de nouveaux systèmes de sécurités.

Les grandes entreprises possèdent les moyens de se protéger, mais les petites et moyennes entreprises n'investissent que peu dans la sécurité du réseau d'entreprise, faute de moyens ou de connaissances dans le domaine.

En effet, un système de sécurité et de surveillance du réseau a émergé : Le *Intrusion Detection System* (IDS). Cet outil permet de surveiller le réseau, car l'analyse de ce qu'il s'y passe est aussi important que sa défense.

Snort est un Network Intrusion Detection And Prevention System, permettant de surveiller le réseau d'entreprise, mais aussi d'avoir une interaction avec ce dernier, pour agir en fonction d'une possible cyber-attaque.

Figure 1 : Snort surveillance



(Ilir Kadriu)

Snort est un logiciel libre et très répandu dans le monde professionnel, c'est pourquoi il sera étudié et mis en œuvre lors de ce travail de recherche, qui a pour but de montrer l'importance de la surveillance d'un réseau d'entreprise, comment le mettre en place avec Snort, et aussi, comment aider une petite ou moyenne entreprise à automatiser la surveillance de son réseau à l'aide d'un fournisseur de sécurité.

Table des matières

Liste des tableaux

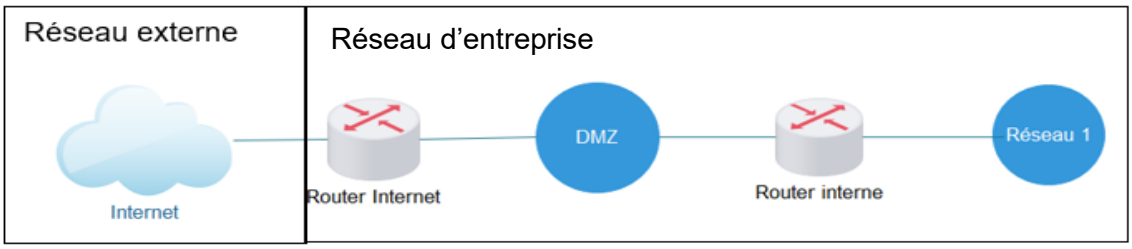
Liste des figures

1. Introduction

Dans une entreprise, les notions de sécurité et de protection de l'information sont essentielles. En effet, plus la force d'attaque sur le réseau se développe, plus on doit développer la sécurité et la surveillance du réseau.

Il y a une distinction à ne pas négliger entre le **réseau d'entreprise** et le réseau externe qui représente **Internet**. Les données transitent de l'entreprise vers l'extérieur et vice-versa. De ce fait, on doit se protéger des intrusions malveillantes dans le réseau d'entreprise, mais aussi surveiller le trafic à l'intérieur du réseau, pour prévenir des possibles attaques internes.

Figure 2 : Réseau interne et externe



(Ilir Kadriu)

On peut voir sur la figure 2 la différence entre notre réseau d'entreprise et le réseau externe de l'entreprise, qui est considéré comme Internet, ou bien le réseau cantonal (RC).

La DMZ (zone démilitarisée) est la partie du réseau qui contient les différents serveurs en production accessibles depuis le réseau externe.

Il faut donc mettre en place des politiques de sécurité pour la surveillance des flux de données qui transitent dans le réseau d'entreprise, pour protéger ces données et les services d'une entreprise pour que celles-ci restent intactes.

De plus, la sécurité des services et des données d'une entreprise diffère vis-à-vis de la taille de l'entreprise. Effectivement, une PME ne va pas avoir les mêmes moyens techniques et informatiques, ni même une gestion de la sécurité, alors qu'une grande entreprise va, dans la plupart des cas, avoir un département qui s'occupe de ce cas.

Lors de ce travail, nous allons nous concentrer sur la surveillance du réseau d'entreprise, car la sécurité de réseau est un sujet vaste, il est donc important de se fixer sur un point à étudier qui est la détection d'intrusion par la surveillance du réseau.

Il est important d'avoir une solution à fournir qui soit peu onéreuse, et qui puisse être facilement configurable. Le but étant que le réseau d'une PME puisse être surveillé et que le fournisseur de sécurité apporte son aide au client pour faciliter ce travail de surveillance et qu'il ait le moins de configuration à faire du côté du client.

Lors de ce travail, nous étudierons aussi la manière de mettre à jour les politiques de sécurités au niveau de la surveillance, au fur et à mesure, toujours dans le but d'avoir une mise à jour qui soit facilitée par le fournisseur de sécurité, dans la ligne directrice d'exporter la gestion de la surveillance d'une PME.

De plus, ce travail va démontrer un package de configuration que l'on pourrait fournir à une PME, avec les démarches à suivre et les moyens mis en œuvre afin de garantir cette surveillance du réseau. Nous allons donc voir les différents chapitres que l'on va aborder lors de ce travail :

- Dans le chapitre deux, nous allons étudier :
  - Ce qu'est un IDS
  - Le fonctionnement d'un IDS
  - La différence entre HIDS et NIDS
  - Les avantages et inconvénients des HIDS et NIDS
  - Ce qu'est un IPS
  - Les différences entre IPS et IDS
  - Les divers outils de protection, du Firewall à l'Access-List et leur comparaison avec un IDS
  - Le choix d'un outil ou de plusieurs outils pour notre PME
- Dans le chapitre 3, nous allons nous concentrer sur l'IDS Snort, nous allons donc analyser :
  - La fonction de Snort
  - Son utilité dans la surveillance de réseau
  - De quoi est composé Snort
  - Ce qu'est une règle Snort
- Dans le chapitre quatre, nous allons nous concentrer sur mon apport personnel, c'est-à-dire :
  - La configuration de Snort
  - L'exemple d'écriture des règles spécifiques de surveillance
  - L'automatisation de la gestion de mise à jour des règles Snort.
  - Le lancement de Snort
- La conclusion :
  - Une synthèse générale du travail
  - Ce que j'ai appris en effectuant ce travail

- Ce que l’on pourrait faire par la suite pour développer certaines questions ou étendre le sujet.

2. Les outils de protection

Dans une entreprise, il est important de poser un constat : Tôt ou tard, il y aura une tentative d'intrusion / une attaque. Le risque d'attaque est toujours présent, c'est pourquoi il faut trouver des politiques de sécurité à appliquer et donc les outils nécessaires pour mettre en œuvre ces directives.

2.1 Le choix de la surveillance du réseau

À travers le vaste sujet qu'est la sécurité d'entreprise, nous allons nous concentrer sur la surveillance du réseau d'entreprise par un IDS (Intrusion Detection System). Il est primordial de le stipuler, car il y a de nombreux sujets dans le thème de la sécurité d'entreprise, et la surveillance est une partie très importante.

Une entreprise possède des données et ces dernières sont précieuses. En effet, les vols de données peuvent avoir plusieurs conséquences pour une entreprise :

- Perte de clients
- Image de l'entreprise dégradée
- Perte financière
- Faillite potentielle

C'est pourquoi il est important d'avoir des politiques de surveillance du réseau d'entreprise.

2.2 Qu'est-ce que la surveillance ?

Il est important de noter que nous allons souvent citer le terme « signature » de paquet lorsque nous parlerons de surveillance c'est pourquoi nous allons immédiatement le définir :

- **Signature d'attaque** : Une signature d'attaque va exprimer ce que va faire l'attaque, sa composition et ses caractéristiques. Ce sont les empreintes de paquets malveillants. Un antivirus et un IDS possèdent une base de signatures d'attaques qui vont leur permettre de reconnaître si le paquet correspond à une attaque ou non. Par exemple, les chaînes de caractères qui représentent la partie d'une attaque seront une signature, en d'autres termes, son empreinte.

On peut aussi indiquer que l'IDS va utiliser différents patterns pour fonctionner :

- Le pattern Matching : L'IDS va comparer les chaines de caractères des paquets à des signatures d'attaques qu'il possède dans sa base. Les attaques doivent donc être représentées sous forme de signatures dans la base de l'IDS.

2.3 Les outils de sécurité que nous allons décrire

Nous allons voir plusieurs outils, les comparer pour avoir toutes les cartes en mains afin de choisir le(s) meilleur(s) composant(s) du réseau pour répondre à nos politiques de surveillance d'entreprise. Nous verrons donc :

- L'IDS
- Le Firewall
- L'IPTables
- L'Access-List

J'ai décidé de choisir ces outils car ce sont ceux-là auxquels on pense en premier lorsqu'on parle de surveillance du réseau.

2.4 IDS (Intrusion Detection System)

Premièrement, il sera question d'IDS, car depuis un certain temps, les IDS sont des outils très répandus et très utilisés dans diverses entreprises. En effet, l'IDS est utilisé car il alerte d'une éventuelle attaque qui se déroulerait au sein du réseau. On peut donc se fier à cet outil pour nous informer des éventuels événements qui se déroulent sur notre réseau.

- Au niveau des entreprises, on peut constater que Swisscom utilise un IDS nommé « Managed IDS/IPS »[11] . Grâce à cet outil, Swisscom va analyser le trafic, avertir et enregistrer les tentatives d'attaques, qui seront lues plus tard par un analyste de la sécurité.
- Il y a aussi des IDS qui sont des logiciels libres comme « Suricata »[22], qui permettent de surveiller notre réseau en temps réel.
- Nous allons nous concentrer sur Snort : un IDS qui va nous permettre de surveiller notre trafic et nous informer des potentiels intrus sur le réseau.

2.4.1 Les signatures d'attaques dans un IDS

Nous allons donc analyser plus spécifiquement comment un IDS agit avec les signatures de paquets. La base de signatures est quasiment la même du côté de l'antivirus et du côté de l'IDS, cependant pour un IDS open-source, on peut modifier et/ou ajouter des signatures qui seront plus spécifiques à la surveillance. Il est important que la base de signatures soit mise à jour régulièrement, pour enregistrer les nouvelles signatures d'attaques.

Il est également important de préciser aussi ce qu'est une « anomalie » dans le cadre d'un IDS :

---

[1] : Site référence : http://mss-i.swisscom.ch/fr/offre/managed-ids/

[2] : Site référence : https://suricata-ids.org/

- Une **anomalie** est une activité de paquet inattendue. C'est-à-dire qu'un paquet ayant certaines caractéristiques ne va pas contenir ce que l'on pense. On peut donc lier les attaques à l'anomalie de paquet, car un simple paquet de données transitant dans le réseau ayant des caractéristiques qui ne présentent pas de danger immédiat, peut avoir un comportement anormal et donc ce paquet pourrait être une attaque. Pour détecter les anomalies, il faut être très vigilent, car parfois ces anomalies sont très subtiles.

Une anomalie est détectée par le pattern-matching, qui va comparer les signatures de paquets avec ses signatures de bases.

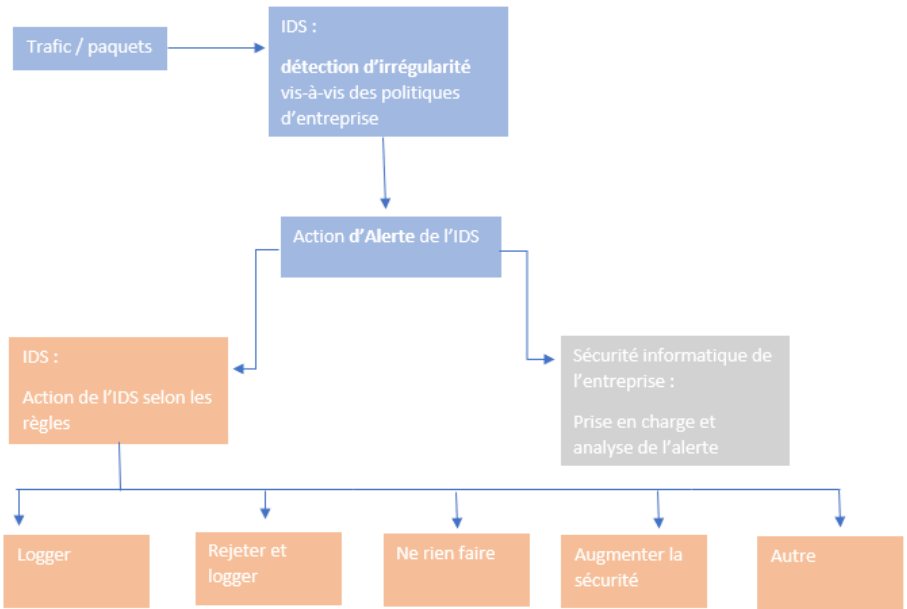
2.4.2 Comment fonctionne un IDS

Dans un schéma réseau, il y a du trafic illustré par des paquets regroupés dans les trams. La protection des données est une politique qui incite à surveiller ce trafic, c'est pourquoi il faut aborder le thème de l'IDS.

Un IDS est un système qui permet de surveiller le trafic dans le réseau. Ce système va détecter les anomalies des activités des trames qui se dirigent de l'extérieur à l'intérieur du réseau d'entreprise, ou bien des trames qui représentent des anomalies de manière interne au réseau de l'entreprise.

Voici le fonctionnement d'un IDS et ses options possibles :

Figure 3 : Fonctionnement de l'IDS



(Ilir Kadriu)

Sur la figure 3, on voit le trafic de paquets qui est observé par l'IDS. Ce dernier va repérer les événements irréguliers, les anomalies, vis-à-vis de sa base de signatures et va déclencher une action d'alerte. L'alerte, d'un côté, sera prise en charge par la sécurité informatique de l'entreprise, et de l'autre, elle pourra déclencher une autre action via l'IDS (par exemple une alerte).

Ce schéma représente notre configuration d'IDS que nous verrons plus tard avec Snort.

Un IDS peut être de types différents :

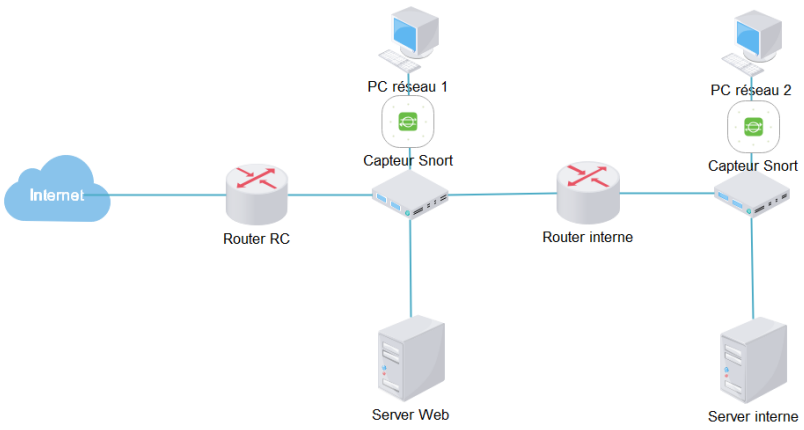
- HIDS : Host Intrusion Detection System
- NIDS : Network Intrusion Detection System

2.4.3 HIDS (Host Intrusion Detection System)

L'HIDS un système de détection d'intrusion qui est basé sur un (ou des) hôte(s) du système. Il est installé de manière à ce qu'il puisse repérer une intrusion suspecte. De ce fait, il va enregistrer l'évènement et ensuite faire une notification à l'hôte. L'HIDS va donc se positionner devant l'hôte et va analyser s'il y a quelque chose ou quelqu'un qui essaie de contourner la politique de sécurité du système de l'hôte.

Ce système de détection installé sur l'hôte va analyser le trafic qui arrive et se présente comme ci-dessous :

Figure 4 : Fonctionnement du l'HIDS



(Ilir Kadriu)

Dans ce schéma, on peut distinguer deux hôtes du système. Ces deux PC ont un HIDS installé séparément qui va surveiller les activités de l'hôte et fournir des logs et des traces d'audits de l'OS s'il détecte des signatures anormales par rapport à sa base.

On peut voir que sur un réseau donné, il y aura autant de HIDS que d'hôtes réseau. Évidemment, le Host Intrusion Detection System présente des points forts et des points faibles :

Concernant ses points forts :

- Il faut constater que le HIDS va être beaucoup plus précis que le NIDS car il analyse, via les logs et via les traces audits, les activités qui se déroulent seulement sur sa machine.
- L'HIDS est plus à l'écoute et à l'affut de ce qu'il se passe chez son hôte, donc il pourra par conséquent réagir plus rapidement.

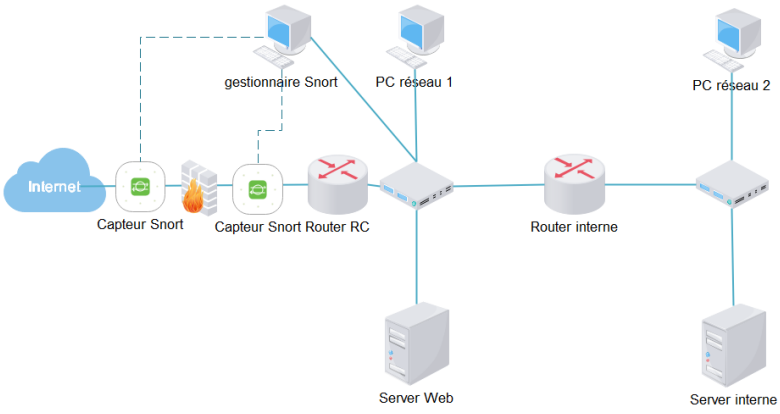
Cependant, cette sensibilité va aussi amener des points faibles :

- Vu qu'un HIDS va engendrer beaucoup de données (logs et traces d'audits), ce système va être dans l'incapacité de faire face à certaines attaques comme une attaque de DOS qui pourrait justement inonder les logs de l'hôte et donc rendre caduque ce fichier de logs.
- Les HIDS doivent être gérés et configurés par l'hôte, donc la gestion de la surveillance dans un réseau à N* hôtes est difficile à envisager.
- Il n'y a pas de système centralisé, c'est-à-dire que lorsqu'il y a un changement de configuration de HIDS sur chaque poste, il faut donc modifier toutes les configurations sur chaque poste, ce qui est une perte de temps.

2.4.4 NIDS (Network Intrusion Detection System)

Network Intrusion Detection System est un système basé sur l'IDS qui va placer des capteurs, donc des cartes réseaux que l'on va placer dans des périphérique réseaux, pour qu'ils puissent analyser les flux entrants et sortants du réseau**.**

Figure 5 : Fonctionnement d'un NIDS



(Ilir Kadriu)

Dans cette figure 4, on peut voir des capteurs NIDS qui peuvent se placer avant le Firewall, ou après.

Nous verrons ce qu'est un pare-feu et sa fonction dans la suite du travail.

2.4.4.1 Capteur NIDS

Un capteur NIDS est un moyen de capter le trafic à un point donné dans le réseau. Il doit être indétectable, c'est pourquoi on va le munir d'une adresse IP qui est : 0.0.0.0.

Les capteurs de NIDS sont généralement reliés à un autre matériel de réseau par une carte réseau. Effectivement, l'intérêt de l'ajouter en complément à autre matériel réseau (par exemple en complément du Firewall ou autre), permettrait de ne pas utiliser plus de périphériques réseau. On va donc configurer cette carte pour qu'elle soit indétectable à travers le réseau par des potentielles attaques.

Dans cet exemple, on place les capteurs à des endroits stratégiques du réseau, celui qui est avant le pare-feu va avoir un contrôle sur tout le réseau, y compris le serveur WEB et va permettre de surveiller les potentielles attaques qui n'ont pas été détecté par le Firewall. Celui qui est après le pare-feu va surveiller tout le réseau qui transite jusqu'à l'internet, cependant il va donc avoir beaucoup plus de flux de paquets à analyser. On va décider de placer les capteurs NIDS selon les besoins de l'entreprise. Le NIDS va détecter le trafic passant sur le réseau.

2.4.4.2 Points positifs et négatifs des NIDS

Contrairement au HIDS, ce système de détection réseau va être plus facilement gérable, car c'est sur une seule console du réseau que l'on va gérer les évènements à surveiller et que l'on va appliquer la politique de sécurité de surveillance.

Le NIDS présente des points positifs et des points négatifs :

Concernant ses points positifs :

- Il va analyser l'entièreté du trafic sur le réseau, et donc pourra être plus étendu et aura une vue globale du trafic de flux de paquets.
- Le NIDS ne va pas utiliser la bande passante de manière à la polluer, car il va seulement analyser les signatures suspectes.
- Il va permettre d'agir en temps-réel.
- Le choix de l'emplacement des capteurs est flexible, on peut les placer où on le souhaite dans le réseau et surtout, l'emplacement est flexible par rapport aux besoins d'une entreprise.

Concernant ses points négatifs :

- Il va analyser le trafic sur le réseau, mais il ne sait pas ce qui va se passer sur la machine de l'hôte du réseau par la suite, c'est l'HIDS qui va s'occuper de cette partie.
- Il faudra ajouter des capteurs, donc des cartes réseaux, sur des périphériques comme un Switch ou bien un Firewall, donc ça peut être un coût pour une entreprise.

2.4.5 IPS (Intrusion Prevention System)

Un IPS se place derrière le pare-feu et va faire la sélection négative du contenu qui peut être malveillant. Contrairement à l'IDS, l'IPS va directement agir en mettant des actions de manière automatique. L'IPS va supprimer/bloquer le trafic (les paquets) malveillant directement, ou alors bloquer le trafic depuis l'adresse source.

L'IDS est perçu comme une surveillance et une alerte du réseau, alors que l'IPS va concrètement agir et peut prendre le risque d'éliminer des paquets qui ne sont pas des attaques ou des paquets malveillants.

Lorsque nous parlons d'agir, nous entendons évidemment avoir une interaction directe avec le réseau. Effectivement, l'IPS, par ces actions comme le rejet de paquet ou autre, va interagir directement avec le réseau, alors que l'IDS va surveiller mais n'aura pas d'impact direct avec le réseau. L'IPS est actif dans l'interaction avec le réseau, tandis que l'IDS est passif dans cette interaction.

2.4.6 Points forts et points faible des IDS

L'IDS est un outil de sécurité qui présente des points forts et des points faibles, dépendant de l'usage que l'on en fait. Voici un aperçu de ses caractéristiques :

Concernant ses points forts :

- Une surveillance précise et forte du réseau d'entreprise pour le NIDS
- Une surveillance précise et forte des activités de chaque hôte pour le HIDS
- Une flexibilité de l'architecture à mettre en place, c'est-à-dire que nous pouvons placer nos capteurs comme bon nous semble, et s'il y a un changement du système de réseau, nous pourrons modifier l'emplacement de nos capteurs.

Concernant ses points faibles :

- Il faut avoir une certaine connaissance du réseau pour utiliser et configurer un IDS.
- La gestion de l'HIDS doit se faire sur chaque hôte du réseau
- Un IDS n'est pas suffisant pour protéger un réseau d'entreprise à lui-seul.

Concernant ce dernier point négatif, nous allons donc maintenant décrire différents outils de protection dans une entreprise, comment ils fonctionnent, où ils se placent et nous verrons la comparaison avec l'IDS.

2.5 Autres outils de sécurité

Mis à part les IDS, il y a un certain nombre d'outils très utilisés et indispensables dans la protection et la sécurité d'une entreprise.

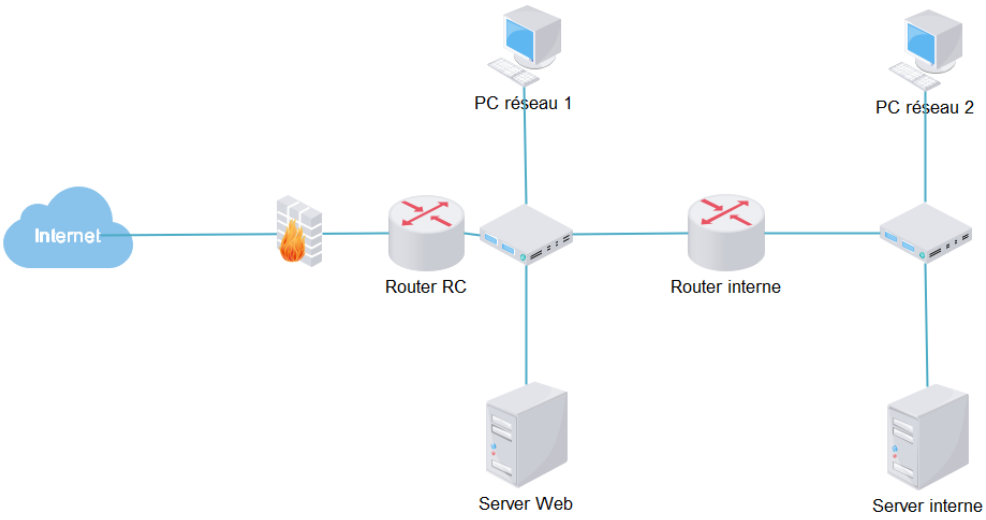
Dans ce sous-chapitre, nous allons décrire et comparer :

- Le Firewall
- L'Access-List
- L'IPTables (sous Linux)
- Le choix d'outils adéquats

2.5.1 Le Firewall (pare-feu)

Un Firewall est un outil utilisé dans quasiment toutes les entreprises pour contrôler le trafic entrant/sortant de notre réseau. Il va décider d'autoriser ou de bloquer les paquets qui transitent dans le réseau via plusieurs critères comme l'adresse IP source, destination, ou autre élément. Il se place d'habitude à la frontière du réseau d'entreprise et du réseau externe (internet) comme présenté ci-dessous :

Figure 6 : Fonctionnement d'un Firewall



(Ilir Kadriu)

Il est important de souligner que le Firewall sur ce schéma, est un pare-feu matériel, donc un hardware, à différencier des Firewall qui sont normalement disponibles dans le système de sécurité des PC.

L'avantage du Firewall matériel, c'est qu'il est complètement indépendant des hôtes et donc qu'il peut bloquer l'accès à tout le trafic réseau et non pas seulement au trafic malveillant qui survient sur un hôte du réseau. Le pare-feu est donc un mur qui va décider du trafic entrant dans le réseau via la signature de ses paquets, selon la configuration qui lui a été implémentée.

Le Firewall fonctionne avec une base de signatures d'attaques contenant le type de protocole du paquet, l'adressage IP, les ports utilisés, et il fonctionne aussi avec l'état TCP (ou bien UDP, donc le protocole de transfert utilisé) du flux de paquet, c'est-à-dire qu'il ne va pas seulement analyser paquet par paquet, mais par un flux, donc il garde en mémoire l'état du paquet qui précède celui qu'il est en train d'analyser.

2.5.1.1 Comparaison avec un IDS

Si on regarde la comparaison qui peut être faite entre un IDS et un Firewall, on peut voir que le Firewall a des actions différentes de l'IDS :

- Le Firewall va autoriser ou bloquer le paquet de données selon la signature totale d'un paquet, c'est-à-dire : le type de protocole, l'adresse IP de destination, l'adresse IP source et les ports utilisés.
- L'IDS va analyser le trafic passant, alerter, enregistrer les logs ou bloquer les paquets selon plusieurs critères et plusieurs signatures (totales ou partielles)
- L'IDS n'aura pas non plus besoin de la signature intégrale d'un virus ou d'une entité malveillante, une partie seulement de la signature va lui suffire pour détecter l'anomalie et alerter, et c'est pourquoi il est considéré comme « intelligent ».
- La liste des signatures d'un IDS est mise-à-jour automatiquement en ligne, vis-à-vis des nouvelles signatures d'attaques (fonctionnement que l'on verra au chapitre 3)

À travers un système de détection d'intrusions, on va analyser le trafic, et voir ce qu'il se passe sur le réseau au lieu d'autoriser ou bloquer le trafic de manière linéaire. Il faut savoir que le Firewall bloque certaines attaques, mais si un IDS peut être utilisé en complément d'un Firewall pour augmenter la sécurité réseau et respecter les politiques de sécurité d'une entreprise. Il est préférable qu'ils soient utilisés simultanément.

Si on fait une analogie avec l'IDS et le Firewall : Le Firewall est un videur, l'IDS est une caméra de surveillance (ou plusieurs dans le cas de plusieurs capteurs NIDS ou plusieurs HIDS dans le réseau). Le Firewall va faire le filtre, ensuite l'IDS va analyser si le filtre a bien été fait, et alerter en cas de flux qui est considéré anormal.

2.5.2 ACL (Access-list)

L'Access-list est un outil de sécurité qui se place sur un ou plusieurs routeurs. Il permet d'avoir un contrôle du trafic du réseau. En effet, cette liste va nous permettre de filtrer les paquets dont l'adresse IP source ne correspondant pas à la configuration indiqué dans notre liste.

Il est important de souligner que l'ACL ne fonctionne pas avec l'état TCP d'un paquet contrairement au Firewall. C'est-à-dire qu'il va seulement se concentrer sur ce qu'il observe, paquet par paquet, sans avoir en mémoire ce qu'il s'est passé précédemment avec les autres paquets dans le flux.

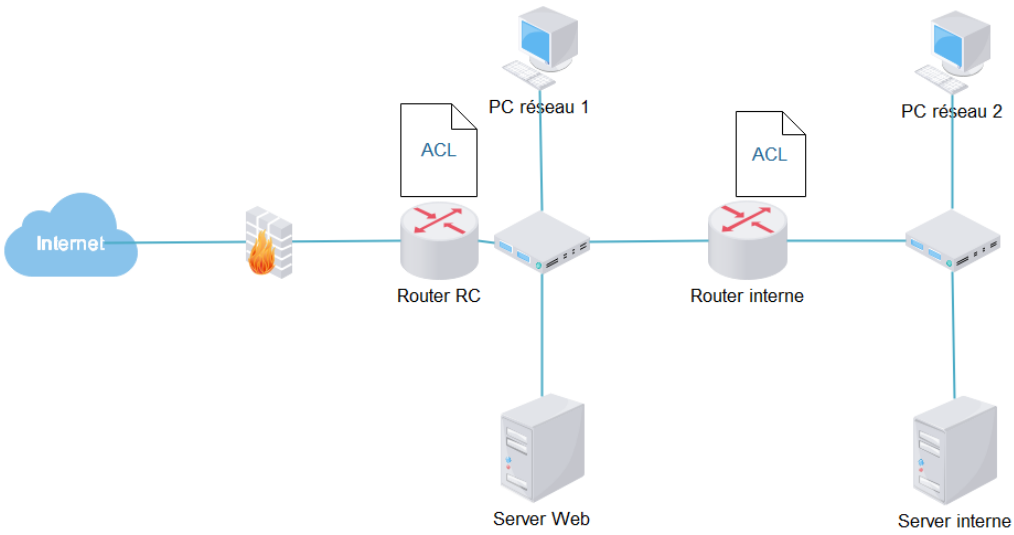
L'ACL va restreindre l'accès au réseau d'entreprise ce qui va permettre d'avoir une certaine sécurité.

On va décider de filtrer selon certains critères :

- Les adresses IP de destinations ou de sources
- Les protocoles

Ci-après, voici ou se place l'Access-list dans un schéma réseau donné :

Figure 7 : Fonctionnement d'une Access-list



(Ilir Kadriu)

Sur la figure 7, on peut voir que l'Access-List est présente dans les deux routeurs, le premier qui fait le lien avec Internet (le réseau cantonal pour notre cas plus précisément) et le second qui est dans le réseau interne.

Pour voir comment fonctionne une Access-List, voici un exemple de configuration d'Access-List sur un routeur :

- Routeur(config)#access-list 1 permit 192.168.1.0 0.0.0.255

Dans cet exemple, on configure une ACL simple. L'access-list 1 va simplement autoriser l'accès à son réseau via l'adresse 192.168.1.0 0.0.0.255.

- Routeur(config)#access-list 101 permit tcp 192.168.1.0 0.0.0.255 172.16.1.0 0.0.0.255 80

Dans cet exemple de configuration, on configure une ACL étendue. L'access-list 101 va autoriser les paquets contenant l'adresse IP source dans le réseau indiqué : 192.168.1.0 0.0.0.255 et seulement par le protocole TCP qui se dirige donc vers le réseau indiqué, c'est-à-dire : 172.16.1.0 0.0.0.255.

2.5.2.1 Comparaison avec un IDS

Voici plusieurs différences entre l'ACL et l'IDS :

- Les capteurs de réseau NIDS peuvent se trouver à plusieurs endroits stratégiques du réseau permettant une surveillance accrue du trafic.
- La gestion de l'IDS (NIDS) peut être centralisée et donc on peut gérer les signatures de comparaison et les règles sur un seul hôte surveillant. On va pouvoir gérer dynamiquement le trafic.
- L'ACL doit être modifiée manuellement à chaque fois sur le routeur, pour ajouter/supprimer des adresses.
- L'Access-list ne surveille pas intelligemment : elle bloque ou autorise des paquets selon l'IP source mais ne regarde pas les signatures des paquets. Elle ne fait que regarder l'adresse IP source avec le port et l'adresse IP destination, donc un paquet contenant un script malveillant pourrait très bien arriver sur le réseau s'il fait partie d'un réseau accepté par l'access-list.
- L'IDS va analyser plusieurs critères des paquets cités dans le chapitre IDS, pour mener la surveillance.

En synthèse, on peut constater que l'ACL est un outil que nous allons configurer de façon manuelle sur chaque routeur. Il ne permet pas non plus de faire une analyse de ce qu'il se passe, mais il va simplement appliquer le permit ou deny des adresses IP indiquées dans sa liste, alors que l'IDS va, quant à lui, analyser un flux de données avec la signature des paquets qui transitent et avec des règles et signatures mises à jour régulièrement et automatiquement, surveiller le réseau, puis agir en fonction de ce qui se passe.

2.5.2.2 Comparaison avec un Firewall

La grande différence entre le Firewall et l'Access-List :

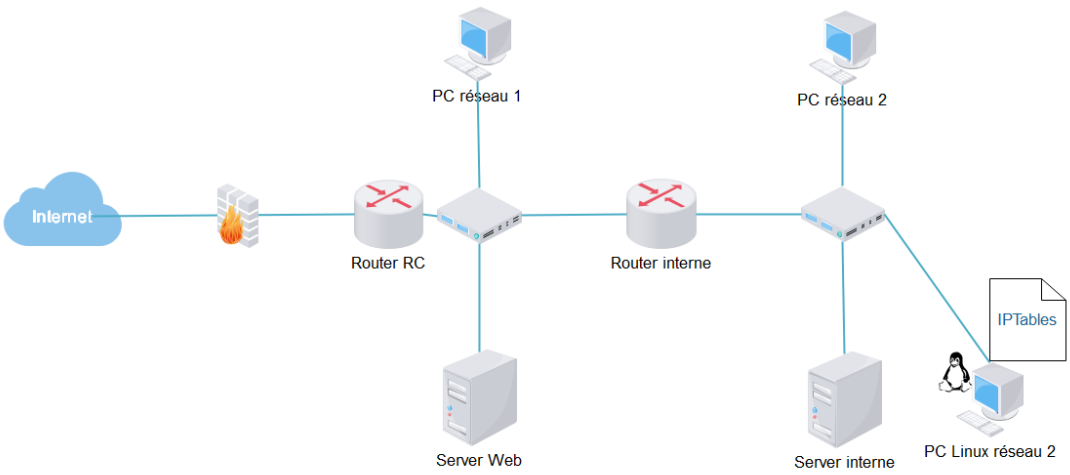
- Le Firewall s'occupe d'inspecter le trafic par flux (plusieurs paquets à la suite avec l'état en mémoire du précédent) alors que l'ACL s'occupe d'inspecter paquet par paquet.

2.5.3 IPTables

IPTables est un logiciel disponible sur GNU/Linux. C'est un outil qui est utilisé pour configurer Netfilter, c'est une infrastructure qui représente le Firewall de Linux. Avec IPTables, on peut filtrer un paquet vis-à-vis de son adresse de destination, son adresse source et le port.

On va donc le placer dans le Firewall comme ci-dessous :

Figure 8 : Fonctionnement de IPTables



(Ilir Kadriu)

On peut donc voir sur cette figure 8 que IPTables se configure sur un hôte interne Linux.

IPTables se configure avec des règles à ajouter à une chaine de règles. Dans ces règles, on retrouve certaines actions qui seront semblables à certains IDS (que l'on verra au chapitre 3) mais ces règles ne sont pas utilisées pour les mêmes besoins.

Nous allons voir un exemple d'ajout de règle de configuration d'une IPTables :

- Linux# Iptables -A INPUT -p udp --source 192.168.1.1 -J DROP

Avec cette commande, on va ajouter à la chaine de règles, une règle qui permet de refuser le paquet ayant comme adresse source 192.168.1.1 et comme protocole UDP.

2.5.3.1 Comparaison avec un IDS

Voici plusieurs différences entre un IDS et IPTables :

- IPTables sera à bloquer un accès au réseau d'entreprise, en filtrant les paquets avec l'analyse du protocole et de l'adresse IP source.
- Un IDS va, quant à lui, analyser le flux de paquets, et non pas systématiquement bloquer. L'IDS va aussi faire une analyse plus poussée de la signature du paquet et le comparer à sa base de signature.
- IPTables n'a pas la notion d'alerter le gestionnaire de sécurité de l'entreprise, s'il se passe quelque chose de malveillant dans le réseau, alors qu'un IDS en a le pouvoir et il est conçu pour ce cas-là spécifiquement.
- IPTables n'est disponible que sur l'OS Linux
- Un IDS comme Snort est disponible pour plusieurs OS comme Linux ou Windows.

Cette dernière distinction permet de voir la restriction de IPTables qui n'est disponible que sur l'OS Linux. En effet, si une entreprise passe de l'OS Linux à Windows, il sera impossible d'intégrer IPTables.

2.5.3.2 Comparaison avec le Firewall

IPTables et le Firewall sont deux systèmes assez similaires étant donné que IPTables va configurer Netfilter, qui est le modèle similaire de Firewall Windows, mais sous Linux. Donc IPTables va agir sur le Firewall de Linux, donc ce n'est pas vraiment comparable.

On peut tout de même souligner le fait que l'on va créer nos règles pour IPTables qui vont s'ajouter à une chaine de règles. On peut moduler nos détections de signatures, alors que le Firewall sous Windows possède une base de signatures et qu'il filtre en fonction de cette base.

2.5.4 Ce qu'il est préférable d'utiliser

Evidemment, il serait impossible de conseiller un de ces outils sans connaitre l'entreprise, sa structure réseau et ses différents composants (server Web, server mailing, DMZ etc).

Il est d'autant plus favorable d'utiliser certains de ces outils ensemble pour que la sécurité d'entreprise et les politiques de sécurité de cette dernière soient respectées. De plus, il ne faudrait pas non plus utiliser tous ces outils combinés car le trafic pourrait être ralenti et il y aurait de la redondance de surveillance.

Il faut noter que l'ACL, le Firewall, l'IPS et IPTables sont des outils centralisés. Ils ont pour but de sécuriser le réseau d'entreprise en bloquant le ou les paquets. Dans notre cas, un IDS sera plus utile pour la surveillance du réseau d'entreprise.

2.6 Synthèse

Nous allons nous intéresser au PME (petite et moyenne entreprise), pour lesquels il n'y a pas souvent de département sécurité du réseau et donc voir quels sont les outils et moyens que l'on peut mettre en place pour pallier leurs besoins de gestions de la sécurité out sourcé.

Pour cela, un IDS particulier sera utilisé : Snort. Grâce à la facilité de mise en place et la flexibilité de ses règles de sécurité, cet IDS va permettre de répondre aux politiques de sécurité de surveillance d'une PME. Évidemment, dans une PME, Snort sera mis en place mais ça n'empêchera pas l'entreprise de conserver son Firewall s'il en a un, car il peut être utilisé en complément.

On pourrait aussi utiliser un IPS pour sécuriser la PME, mais notre but premier étant d'alerter le gestionnaire de sécurité, et non de directement agir en bloquant, car s'il s'est avéré qu'un paquet n'était pas dangereux, mais qu'il était, dans tous les cas, considéré comme anormal, il pourrait être supprimé, et donc on pourrait perdre des données qui seraient utiles à la PME.

Il sera aussi important de voir la manière avec laquelle un fournisseur de sécurité pourra aider le client à surveiller son réseau d'entreprise.

3. Snort

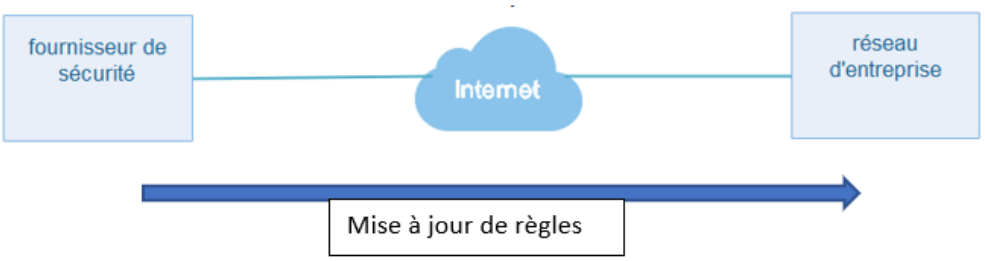
Nous allons étudier Snort, car premièrement, c'est un logiciel libre (open-source), et qu'il est beaucoup utilisé dans les entreprises.

Étant donné que l'on va se concentrer sur la sécurité de réseau d'une petite ou moyenne entreprise, on va mettre en place un système qui permet de surveiller le trafic et de mettre à jour les règles et politiques de sécurité depuis un gestionnaire de sécurité (un fournisseur qui va s'occuper de la surveillance) car une PME n'a peut-être pas les moyens de se munir d'un département « sécurité du système d'information informatisé ».

On va donc se concentrer sur Snort, et avec l'aide de certain Framework comme PulledPork, on va pouvoir mettre à jour les règles de sécurités et surveiller le trafic, « à distance ».

Voilà le schéma que l'on veut reproduire :

Figure 9 : Schéma de la mise en place de Snort



(Ilir Kadriu)

On peut voir dans la figure 9, qu'on reste toujours dans le principe d'une solution de sécurité simple à intégrer, facilement déployable, peu onéreuse et qui est gérée depuis le fournisseur de sécurité.

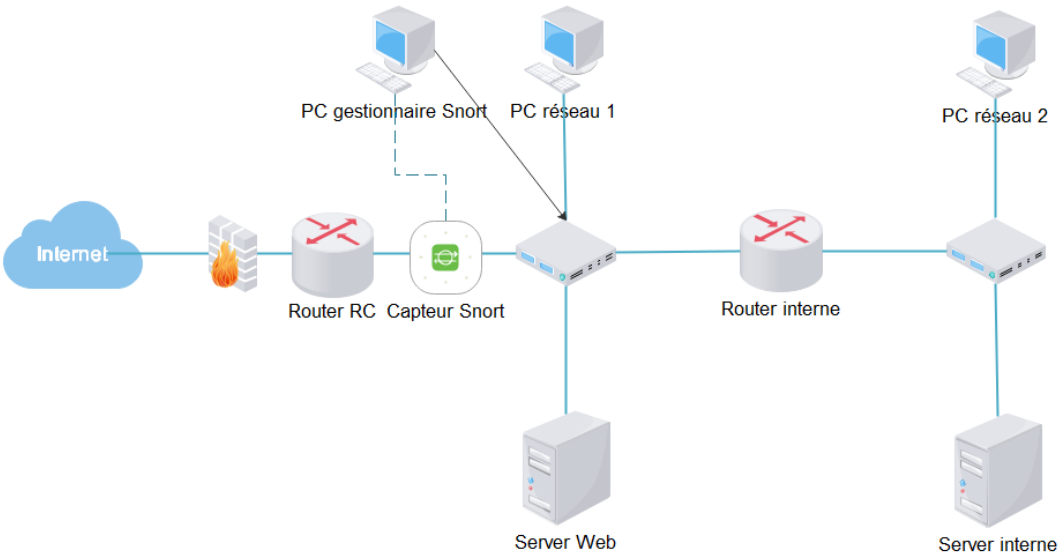
3.1 Qu'est-ce que Snort ?

Snort est un hybride d'IPS et d'IDS utile et efficace. Cet outil va permettre d'analyser le trafic entrant et sortant du réseau. Snort fonctionne en mode Network Intrusion Detection System, et par analyser le trafic, on entend : Observer les paquets selon leur signature ou leur signature partielle. Snort possède une base de signatures qui va être mise-à-jour

quotidiennement, via ses règles. Snort va donc agir lorsqu'il va détecter une anomalie dans un ou plusieurs paquets qu'il va surveiller.

Voici comment est représenté Snort sur un schéma réseau :

Figure 10 : Fonctionnement de Snort



(Ilir Kadriu)

On peut donc voir sur la figure 10, que l'on va placer Snort à un endroit stratégique du réseau qui est seulement après le routeur lié au réseau cantonal. Depuis cet endroit, on peut donc surveiller ce que le Firewall a laissé passer comme paquet, car comme nous l'avons vu au chapitre 2, le Firewall peut laisser entrepasser certains paquets. Donc pour ces raisons, il est préférable d'utiliser Snort à cet endroit pour surveiller les potentielles attaques malveillantes.

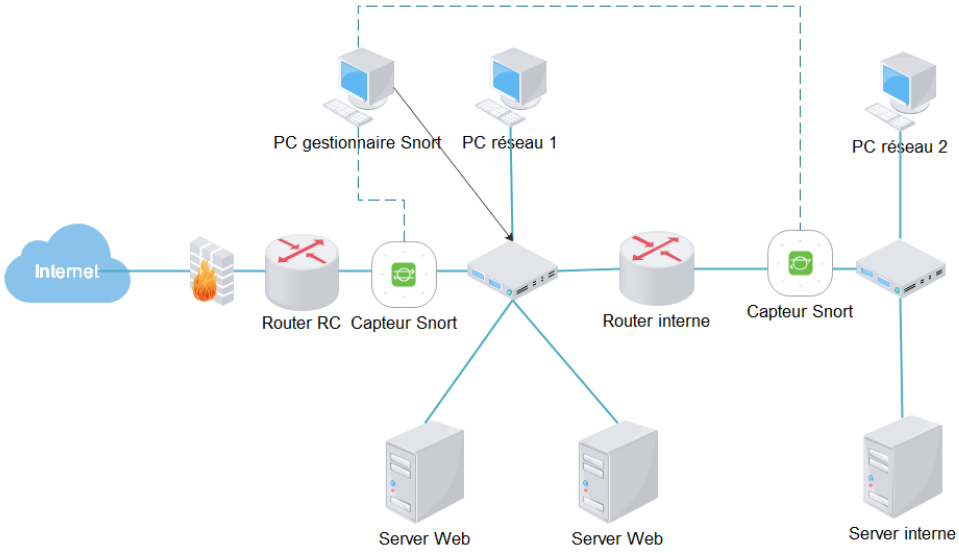
3.2 Que faut-il surveiller dans un réseau d'entreprise ?

Dans notre cas, la PME peut avoir plusieurs données sensibles et plusieurs serveurs à disposition. Un serveur Mailing, un serveur WEB, un serveur Snort, les hôtes du réseau, des bases données et d'autres périphériques. Il faudra établir des politiques de sécurité de surveillance, vis-à-vis de tous les composants du réseau, pour que la surveillance de ces derniers soit complète et optimale.

Il faut noter qu'avec Snort, nous allons être en fonction NIDS, mais nous allons procéder comme un HIDS, c'est-à-dire que Snort sera installé sur chaque hôte du système et va surveiller ce qu'il se passe dans le réseau préalablement défini dans la configuration.

Voici le schéma potentiel du réseau d'entreprise d'une PME :

Figure 11 : Réseau d'entreprise d'une PME



(Ilir Kadriu)

Nous pouvons donc constater dans la figure 11 qu'il y a plusieurs composants comme le serveur Web, le serveur Mailing, et plusieurs hôtes qui se trouvent sur même réseau. Ce sont tous les flux de données qui se dirigent vers ces composants qu'il faudra surveiller, c'est pourquoi nous avons placé les capteurs Snort juste avant le routeur RC car celui-ci va pouvoir surveiller le trafic arrivant dans le réseau d'entreprise, et un capteur juste avant le routeur interne pour que celui-ci surveille ce qui se passe dans la partie privée du réseau. Comme indiqué auparavant, Snort va donc être installé, dans notre cas, sur chaque hôte du système, donc il agit en NIDS, mais se comporte tel un HIDS. C'est un type d'hybride qui va allier les points positifs d'un NIDS et ceux d'un HIDS.

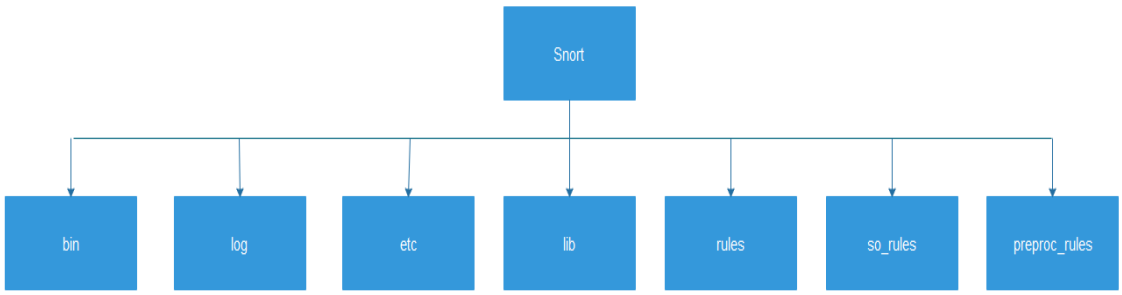
3.3 Comment se compose Snort ?

Snort est un programme qui se compose de plusieurs dossiers où se trouvent ses ressources, son exécutable, ses règles et autres composants.

Nous allons voir en détail de quoi est composé Snort et ce que l'on peut utiliser, modifier, ajouter, ou bien ignorer pour avoir une utilisation optimale de ce programme.

Voici ci-dessous comment se compose Snort lorsqu'on l'installe :

Figure 12 : Composition de Snort



(Ilir Kadriu)

On peut donc voir sur la figure 12 l'arborescence de Snort.

3.3.1 Le contenu des dossiers Snort en détail

Premièrement, il y a le dossier « bin » qui contient l'exécutable Snort et ses composants en type « .dll ».

Ensuite, le dossier « doc » qui contient nombreux fichiers « ReadMe » d'utilisateurs de Snort ayant intégré ces « ReadMe », mais on y retrouve aussi le dossier « signatures » qui va donc contenir les signatures d'anomalies connues.

Par la suite, le dossier « etc » est très important car il contient le fichier « snort.conf » et c'est ce fichier qui va indiquer les path, les constantes, et les variables à utiliser lorsque l'on lance Snort. Nous verrons ce fichier de configuration plus précisément dans l'apport personnel au chapitre quatre.

Ensuite, il y a le dossier « lib » qui va contenir deux dossiers « snort_dynamicengine » et « snort_preprocessor ». Ces deux dossiers contiennent des parcelles d'application lorsque l'on lance l'application. Ces parcelles sont en type « .dll ».

Par la suite, on peut voir qu'il y a un dossier « log ». Ce dossier va contenir les logs qui ont été créés par les événements et les différentes alertes qui ont aussi été générées par les événements.

En ce qui concerne les trois derniers dossiers : « rules », « so_rules » et « preproc_rules », nous allons voir plus en détails leurs contenus, étant donné que ces dossiers représentent le cœur de Snort.

3.3.2 Le dossier de règles Snort

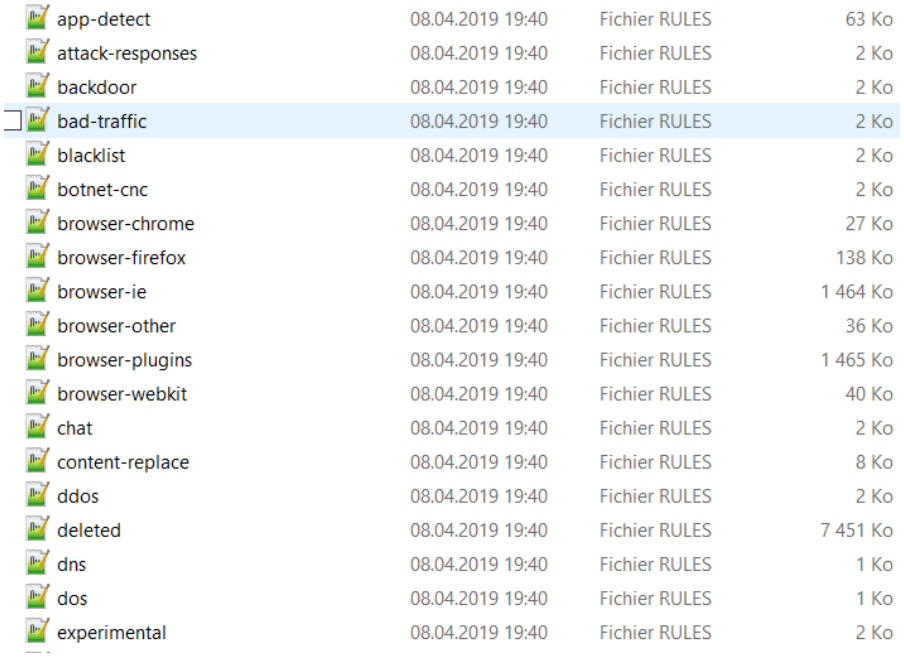
Les règles se trouvent dans trois dossiers. Ces règles sont le cœur de la surveillance Snort, ils vont indiquer ce que l'on doit observer, comment on doit agir, et indiquer des précisions sur les événements. Nous ne verrons pas en détail certains de ces dossiers car on ne va pas tout utiliser, le but étant d'être le plus optimal possible avec l'utilisation du logiciel Snort.

3.3.2.1 Dossier rules

Premièrement, Il y a le dossier « rules » qui est composé de règles Snort. Ce dossier contient les règles que Snort va utiliser pour surveiller le réseau. Nous verrons, dans la suite de ce travail comment se compose une règle Snort.

Le dossier « rules » contient les règles basées sur du texte standard et qui vont être exécutées lorsqu'on lance Snort.

Figure 13 : Contenu du dossier rules



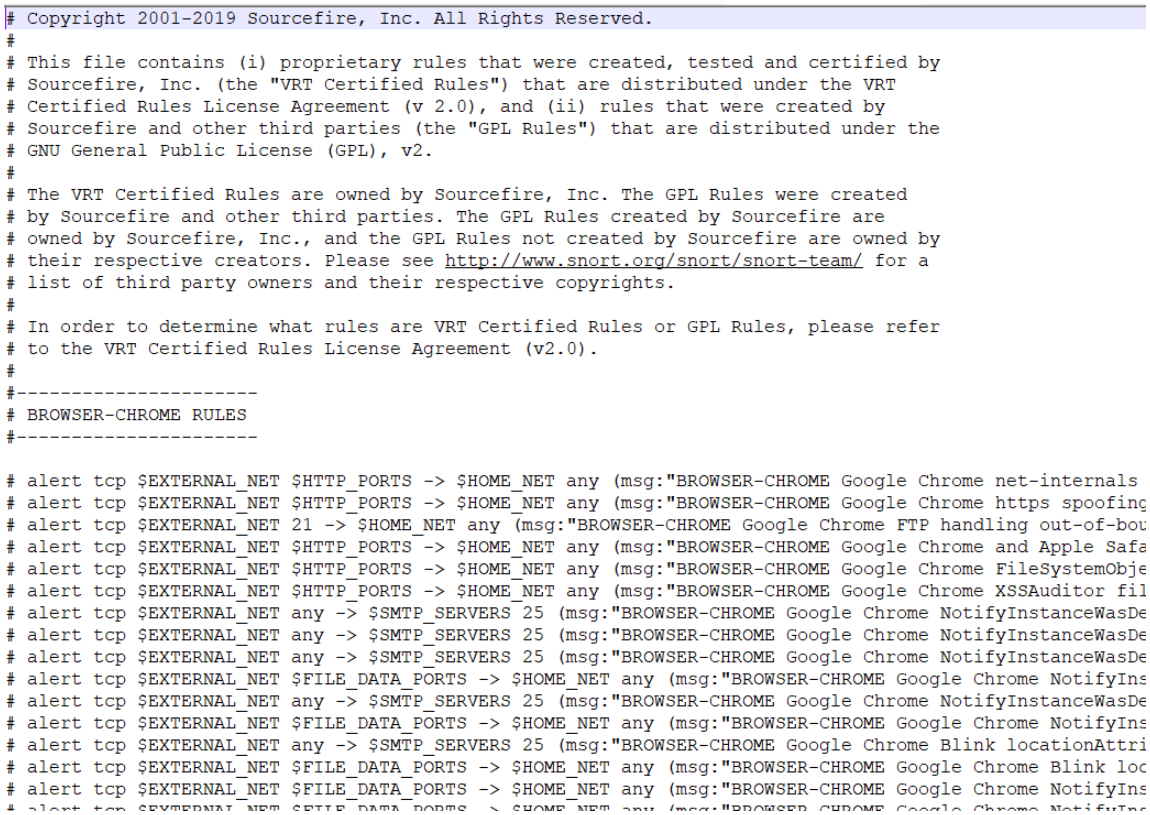
(Ilir Kadriu)

Il est important de noter que le fichier « local.rules » est le fichier qui va être spécifique à chaque hôte du système dans notre cas. Donc il sera mis-à-jour par le fournisseur de sécurité et donc par un expert qui pourra spécifier la règle à implémenter qui sera spécifique à l'entreprise.

Donc on peut voir, sur cette figure 13, que chaque fichier est de type « rules ». Ces fichiers vont contenir des règles Snort qui vont être de diverses compositions et vont être

en relation direct avec le titre du fichier. Par exemple, dans le fichier « browser-chrome », il y aura des règles en lien avec le browser de Google Chrome :

Figure 14 : Contenu du fichier browser-chrome



```
# Copyright 2001-2019 Sourcefire, Inc. All Rights Reserved.
#
# This file contains (i) proprietary rules that were created, tested and certified by
# Sourcefire, Inc. (the "VRT Certified Rules") that are distributed under the VRT
# Certified Rules License Agreement (v 2.0), and (ii) rules that were created by
# Sourcefire and other third parties (the "GPL Rules") that are distributed under the
# GNU General Public License (GPL), v2.
#
# The VRT Certified Rules are owned by Sourcefire, Inc. The GPL Rules were created
# by Sourcefire and other third parties. The GPL Rules created by Sourcefire are
# owned by Sourcefire, Inc., and the GPL Rules not created by Sourcefire are owned by
# their respective creators. Please see http://www.snort.org/snort/snort-team/ for a
# list of third party owners and their respective copyrights.
#
# In order to determine what rules are VRT Certified Rules or GPL Rules, please refer
# to the VRT Certified Rules License Agreement (v2.0).
#
#----------------------
# BROWSER-CHROME RULES
#----------------------

# alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome net-internals
# alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome https spoofing
# alert tcp $EXTERNAL_NET 21 -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome FTP handling out-of-bou
# alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome and Apple Safa
# alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome FileSystemObje
# alert tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome XSSAuditor fil
# alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"BROWSER-CHROME Google Chrome NotifyInstanceWasDe
# alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"BROWSER-CHROME Google Chrome NotifyInstanceWasDe
# alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"BROWSER-CHROME Google Chrome NotifyInstanceWasDe
# alert tcp $EXTERNAL_NET $FILE_DATA_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome NotifyIns
# alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"BROWSER-CHROME Google Chrome NotifyInstanceWasDe
# alert tcp $EXTERNAL_NET $FILE_DATA_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome NotifyIns
# alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"BROWSER-CHROME Google Chrome Blink locationAttri
# alert tcp $EXTERNAL_NET $FILE_DATA_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome Blink loc
# alert tcp $EXTERNAL_NET $FILE_DATA_PORTS -> $HOME_NET any (msg:"BROWSER-CHROME Google Chrome NotifyIns
```

(Ilir Kadriu)

Sur la figure 14, on peut voir le contenu du fichier browser-chrome. On peut donc noter que le port source est souvent de provenance HTTP_PORTS (c'est une variable qui représente une liste de ports qui est déclarée dans le fichier de configuration) et que l'on va spécifier la règle vis-à-vis des caractéristiques du navigateur Chrome.

On peut donc remarquer plusieurs règles qui sont désactivées (on le voit avec le signe #) pour l'instant mais qui peuvent être activées à n'importe quel moment en enlevant le signe « # ».

Nous verrons, tout au long de ce travail, comment se compose une règle Snort, pour mieux comprendre le contenu des dossiers de règles et pour être capable de réutiliser Snort.

3.4 Les règles Snort, les politiques de sécurités/surveillances

Les règles Snort sont écrites pour que les politiques de sécurité par rapport à la surveillance de l'entreprise soient respectées. Cependant, il faut savoir comment écrire

une règle Snort, car il y a des variables, des options et des signatures à rédiger, et nous sommes toujours dans l'optique d'avoir une solution optimale.

Voici la composition d'une règle Snort :

- Rule action
- Protocol
- Source IP Address
- Source Port
- Direction Operator Destination
- Destination IP Address
- Destination Port
- Rule Option

Nous allons donc voir en détail chacun de ces points pour avoir la meilleure utilisation et connaissance d'une règle Snort.

3.4.1 Les actions des Rules

Les règles Snort décrivent une action de base. Il y a plusieurs actions possibles à effectuer sur le trafic :

Tableau 1 : Action de règle

Action	Description
Alerte	l'action va générer une alerte et enregistrer les logs du paquet
Log	l'action va enregistrer les logs du paquet tout simplement
Pass	l'action va ignorer le paquet
Activate	l'action va alerter et ensuite activer une règle dynamique
Dynamic	l'action va rester inactive jusqu'à ce qu'on l'active avec une activate rule, ensuite il agit comme une règle d'action log
Drop	l'action va bloquer et enregistrer les logs du paquet
Reject	l'action va bloquer et enregistrer les logs du paquet. Si c'est un paquet TCP, l'action va envoyer un tcp reset Si c'est un paquet ICMP, l'action va afficher « Destination port unreachable »
Sdrop	l'action va bloquer le paquet mais ne va pas enregistrer les logs comme pour l'action Drop

(Ilir Kadriu)

On peut observer, dans le tableau 1, qu'avec l'action, on peut donc indiquer ce que Snort va faire, par rapport à un événement donné. Soit il va alerter dans le dossier dans le fichier log/alerte, ou enregistrer les logs des paquets avec les informations de l'événement dans le fichier log/log<nombre>, ou le rejeter, etc.

Il est important de noter que seules les règles Alerte, Log, Pass, Activate et Dynamic vont nous être utiles car ce sont ces règles qui vont nous permettre de fonctionner en tant qu'IDS (et non pas en tant qu'IPS car nous allons seulement utiliser la partie IDS de Snort).

Dans la continuité de la règle, on va maintenant décrire les protocoles que l'on peut configurer.

3.4.2 Les Protocoles des Rules

Il faut indiquer à la règle un protocole à surveiller. En effet, il serait très difficile et très lourd en matière de bande passante, d'observer le trafic de tous les protocoles, c'est pourquoi on va spécifier un protocole par règle. Voici le choix des protocoles à disposition :

- TCP
- UDP
- ICMP
- IP

Le protocole UDP aussi appelé « Best Effort » est un protocole de communication et d'acheminement de données, mais sans contrôle d'erreurs.

Le protocole TCP se base sur l'adressage IP et est utile pour la communication entre deux appareils possédants tous deux une adresse IP.

Le protocole ICMP est un protocole qui va nous permettre de faire des transiter des messages sur les machines d'un réseau (par exemple le PING).

Le protocole IP permet le transport de données à travers deux machines possédant une adresse IP et il regroupe ICMP, TCP et UPD.

Nous allons maintenant voir comment gérer l'adressage IP dans une règle Snort.

3.4.3 Les adresses IP sources et destinations des Rules

Pour chaque règle Snort, il faut évidemment une adresse IP source et une adresse IP destination. On peut néanmoins indiquer plusieurs choix, comme une liste d'adresses IP

ou bien même un **any** pour autoriser n'importe quelle adresse IP. Voyons tous les choix possibles :

- Soit on peut utiliser les variable HOME_NET, EXTERNAL_NET en tant que variables (par exemple **ipvar** HOME_NET 10.136.3.0)
- Soit on peut mettre en dur dans le fichier les adresses (par exemple : alert tcp 192.168.0.5 any ->…)
- Soit on peut mettre **any** (pour toutes les adresses)
- Soit on peut mettre gérer l'adressage avec des négations grâce à l'opérateur "!" (par exemple alert tcp !19.168.0.5 any ->…)
- Soit on peut créer une liste en variable dans le fichier « snort.conf » que l'on va utiliser directement dans la règle (par exemple **ipvar** LST_IP [10.136.1.5, 10.136.1.6])

Ces adresses vont nous permettre de définir la source du flux de paquets à surveiller et aussi la destination de ce flux. Il faut aussi garder en tête qu'une attaque ne provient pas forcément du réseau externe à une entreprise, il peut aussi y avoir parfois des intrusions internes à l'entreprise, c'est pourquoi il faut être vigilant avec l'adressage IP.

3.4.4 Les ports sources et destinations des règles

Les ports vont aussi permettre d'optimiser les règles pour ne pas passer par tous les ports. En effet, lorsqu'on veut créer une règle pour la surveillance de « SSH », on sait que l'on va utiliser le port 22 par exemple, car c'est par ce port que les flux de données « SSH » vont transiter.

Pour les ports, on a aussi plusieurs possibilités :

- on peut mettre **any** comme port pour prendre en compte tous les ports
- on peut définir statiquement un port (par exemple **80**)
- on peut définir une liste de ports (par exemple **portvar** PORT_HTTPS [36, 80])
- on peut mettre un port avec une négation (par exemple **!80)**

Donc les ports vont optimiser la règle et son utilisation.

3.4.5 Les opérateurs de direction des Rules

Les opérateurs de directions vont indiquer le sens du trafic à observer. C'est important car, encore une fois, le but n'est pas d'analyser tout le trafic entrant ou sortant, car ça serait trop lourd en bande passante, mais seulement ce qui est susceptible d'être dangereux. Voici les possibilités d'opérateur :

- Dans un seul sens, par exemple **->** ou alors **<-**
- Dans les deux sens donc bidirectionnel : **<>**

Donc pour résumé, on pourrait analyser le flux passant du réseau interne au réseau externe, vice versa, mais aussi dans les deux sens en même temps.

3.4.6 Les options de règles Snort

Nous pouvons indiquer que les options des règles sont, pour la plupart facultatives et pour quelques-unes obligatoires. Elles composent la **signature** de la règle, le contenu de ce qu'il faut analyser dans le paquet. Il est important de voir que nous pouvons indiquer nous-même ce qu'il faudra observer dans le contenu d'un paquet, donc on peut définir manuellement ce que l'on va observer dans un paquet.

Dans une règle, on peut mettre plusieurs options, et celles-ci se divisent en quatre catégories :

- Option générale
- Option PayLoad
- Option Non-PayLoad
- Option Post-Detection

3.4.6.1 Option Générale

Ces options fournissent des informations sur la règle mais n'impactent pas la détection. On y retrouve plusieurs choix :

Tableau 2 : Option générale

Option	Explication	Exemple
Msg	L'option va afficher le message choisi dans le logging et l'alerte	(msg : "Alerte y'a un problème")
Reference	L'option indique quelle est la partie de Snort qui va générer l'événement quand une règle se déclenche, plus précisément, la référence (donc le site) ou Snort a recueilli sa signature.	(gid : "generator id")
Sid	L'option doit être unique et va identifier la règle Snort	(sid : "id choisi pour cette règle")
Rev	L'option est aussi unique et va identifier une révision de la règle Snort.	(rev : "rev Integer")

(Ilir Kadriu)

On peut constater, dans le tableau 2, qu'il faut obligatoirement indiquer l'option « Sid » et l'option « Msg » pour toutes nos règles. Sans cela, on ne peut pas lancer Snort, car on aura une erreur dans le fichier de règles.

Nous pouvons indiquer à l'utilisateur des informations importantes avec ces options générales, ce qui lui permet d'agir en fonction de ces options. Par exemple, dans le fichier log/alerte, il peut voir le message et le sid de chaque événement qui est déclenché par la règle.

Exemple de règle avec des options générales :

- Alerte ICMP $EXTERNAL_NET any -> $HOME_NET any (msg : ''Tentative de ping'' ; sid :55555 ;)

Dans cet exemple, on peut donc constater que les deux options Msg et Sid sont présentes, car ces options sont obligatoires, elles vont identifier la règle avec le nombre 55555 et alerter avec le message « Tentative de ping » lorsque la règle va être déclenchée. On peut aussi voir le protocole ICMP présent, et les variables EXTERNAL_NET et HOME_NET qui sont attribuées à des adresses IP dans le fichier de configuration.

Cette règle ci-dessus va alerter l'utilisateur de toutes tentatives de paquets contenant le protocole icmp de la part d'un hôte qui possède une adresse IP qui fait partie du réseau indiqué par la variable EXTERNAL_NET en direction du réseau indiqué par HOME_NET.

3.4.6.2 Option PayLoad

Les options de PayLoad vont analyser les données dans le PayLoad du paquet et voir si ces données peuvent être liées entre elles. Voici une liste d'option PayLoad :

Tableau 3 : Option PayLoad

Option	Explication	Exemple
Content	C'est une option très importante sur Snort. Elle va permettre à l'utilisateur de chercher à l'intérieur d'un PayLoad le contenu qu'il souhaite.	(content : "POST";)
Rawbytes	L'option permet d'inspecter dans les nouveaux paquets de donnée, en ignorant ce qui a été fait par le pré processor.	(rawbytes;)
Depth	L'option indique le nombre d'octet ou on va chercher le modèle indiqué.	(depth : 5;)
Offset	L'option est comme depth mais on cherche à partir du nombre indiqué.	(offset : 3;)
Http_client_body	L'option va restreindre à rechercher que dans le body d'une requêtes HTTP	(http_client_body;)

Http_cookie	L'option va restreindre la recherche au cookie header d'une requêtes http	(http_cookie;)
Http_header	L'option va rechercher dans le header de la requête HTTP	(http_header;)

(Ilir Kadriu)

On peut observer dans ce tableau 3, que ces options vont donc nous permettre d'analyser le PayLoad des paquets qui transitent et on va pouvoir optimiser l'analyse de ces paquets.

Par exemple, nous allons voir avec l'option « http_header » s'il y a une entité malveillante dans le header de la requête http. Par exemple, une attaque aurait très bien pu être composée d'un script qui se lance depuis le header de la requête, donc on va pouvoir observer cette partie de la requête.

Voici un exemple de règle avec l'option PayLoad :

- Alerte tcp $EXTERNAL_NET any -> $HOME_NET any (msg : ''Tentative de requête POST'' ; sid :55555 ; content : ''POST'' ; http_header ;)

Dans cet exemple, nous pouvons analyser que le contenu est ''POST'' donc nous allons rechercher dans le paquet ce contenu, qui se trouve dans le header seulement, avec l'option http_header.

3.4.6.3 Option Non-PayLoad

Comme son nom l'indique, ces options vont servir à spécifier la règle selon ce qui n'est pas dans le PayLoad. Voici quelques options Non-PayLoad :

Tableau 4 : Option Non-PayLoad

Option	Explication	Exemple
Ttl	L'option va vérifier la durée de vie de l'IP. C'est utilisé pour détecter les tentatives de trace route	(ttl:>5;)
Tos (type of service)	L'option est utile pour vérifier l'IP TOS pour une valeur spécifique	(tos :4;)
Id	L'option est faite pour vérifier l'ID du paquet IP	(id :34423;)
Flags	L'option est utilisée pour vérifier si des flags TCP sont présents	(flags : flagsLetters)
Flow	L'option permet aux règles de s'appliquer juste à certaines directions du flux de trafic	(flow: from_client;) (flow :to_client;) (flow:stateless;)

(Ilir Kadriu)

Dans ce tableau 4, on peut voir que ces options sont très utiles si l'on veut vérifier ce que fait le flux de trafic et comment il est composé. Si par exemple, le flow se dirige vers le client, ou provient du serveur, on va vérifier la durée de vie de l'IP pour voir s'il y a une tentative de « traceroute » de la part d'un paquet qui contient un script malveillant.

On peut voir un exemple d'options Non-PayLoad :

- Alerte EXTERNAL_NET tcp -> HOME_NET any (msg : ''Tentative de requête POST'' ; sid :55555 ; content : ''POST'' ; http_header ;flags ; flow :to_client ;)

Toujours dans notre alerte de header contenant la requête POST, on va ajouter l'option flags pour vérifier si des flags tcp sont présents, et aussi l'option flow, qui analyse le flux qui se dirigent en direction du client.

3.4.6.4 Option Post-Detection

Ces options vont déclencher des règles après la détection d'événements. Voici une liste de quelques options post-detection :

Tableau 5 : Option Post-Detection

Option	Explication	Exemple
Logto	l'option va indiquer à Snort de enregistrer les logs de tous les paquets après une règle spécifique, dans un nouveau dossier log	(logto : LogFolderNew;)
Session	l'option est conçue pour extraire les données d'utilisateurs des sessions TCP	(session: printable;)
Resp	l'option active une réponse qui va tuer la session visée.	(resp;)

(ilir Kadriu)

On peut donc analyser, dans le tableau 5, que grâce à ces options nous allons pouvoir réagir après détection, comme tuer une session que l'on a détecté comme étant anormale, ou bien enregistrer les logs des paquets dans un nouveau dossier de log, dépendant de ce que l'on veut faire dans notre protection de réseau.

On peut voir un exemple d'options Non-PayLoad :

- Alerte EXTERNAL_NET tcp -> HOME_NET any (msg : ‘’Tentative de requête POST’’ ; sid :55555 ; content : ‘’POST’’ ; http_header ;flags ; flow :to_client ; logto : ‘’LogPost’’ ; session: printable ;)

On va donc ajouter l’option logto pour indiquer que nous allons enregistrer les logs dans un nouveau dossier « LogPost ». L’option session va nous permettre d’imprimer dans les logs les informations de la session avec le protocole TCP.

3.4.6.5 Synthèse d’options de règles

Ces options vont permettre de spécifier les règles et ce que l’on surveille. C’est ce qui sera important, car on ne va pas tout inspecter, on va inspecter seulement ce qui pourrait représenter un danger potentiel pour la PME. En effet, on ne va pas non plus polluer la bande passante, donc ces options vont permettre de détecter, par parcelle de signatures ou informations de paquets, s’il y a une anomalie ou un danger potentiel avec les paquets que l’on observe.

3.4.7 Conclusion sur les règles Snort

Pour synthétiser, nous avons donc vu comment fonctionne une règle Snort. Il est important de signaler qu’il faut avoir des règles de surveillance de son réseau, et que celles-ci contiennent les composants et options qui leur permettront d’avoir une surveillance optimale du réseau d’entreprise.

3.5 La mise à jour de règle en temps réel

Il fondamental que les règles soient mises à jour, et que ces mises à jour ne nécessitent pas un redémarrage de Snort, pour qu’on puisse laisser Snort travailler sans interruption de service. Dans notre cas, lorsque le fournisseur de services met en place des nouvelles règles, il souhaite que cela soit transparent pour l’entreprise.

Il est donc important de notifier que Snort prend en considération le changement d'une configuration de règles.

- Il y'a un Thread d'analyse de la configuration
- Il y'a un Thread principal de traitement de paquets.

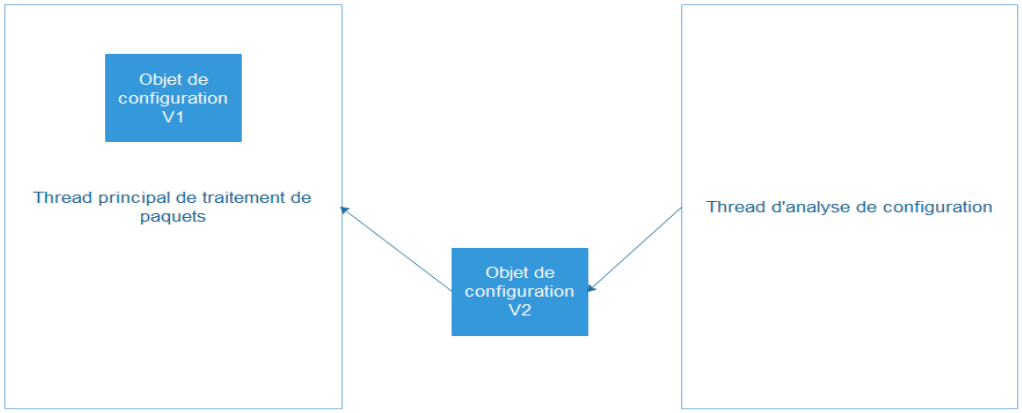
Un Thread est une tâche qui va s’exécuter en parallèle d’autres Thread, donc on peut exécuter deux Threads en même temps avec le même processeur.

Les Threads vont donc fonctionner en parallèle, et le premier Thread d’analyse de la configuration va créer un objet de configuration qui sera utilisé par le thread principal de traitement des paquets.

Lorsqu'il y a des changements dans le Thread d'analyse de la configuration, le thread principal de traitement des paquets va prendre en compte l'objet créé et continuer à s'exécuter dans les nouvelles configurations.

On peut visualiser le fonctionnement de Thread de mise-à-jour Snort ci-dessous :

Figure 15 : Fonctionnement de Thread M-à-J



(Ilir Kadriu)

Comme on peut le voir sur la figure 15, le Thread d'analyse de configuration va créer un objet de configuration contenant les chemins d'accès pour les règles mises à jour, et quand le Thread principal prendra en compte l'objet de configuration version 2, il le remplacera l'ancienne version par celle-ci.

Donc il n'y a pas besoin de relancer Snort pour recharger les règles, ce qui est un point positif dans notre volonté de dynamisme.

3.6 Avancé du travail sur les règles

Dans la continuité de l'apprentissage des règles Snort plus en détail et de manière plus précise, nous pourrions nous intéresser aux règles « so_rules » et « preproc_rules », mais n'étant pas utiles pour notre objectif, nous allons simplement survoler cette partie de règles.

3.6.1.1 Règles so_rules

Le dossier « so_rules » aussi appelé « Shared Object » (SO) est un dossier de règles partagé et grâce à ce dossier, on peut étendre la détection de Snort. Depuis la version 2.6.0, les développeurs de Snort ont ajouté une API pour la détection, qui permet aux utilisateurs de Snort d'étendre les règles avec le langage C.

Pour revenir à notre cas, nous n'allons pas modifier certaines règles pour les rendre plus personnalisées avec le langage C, car il est parfois inutile d'ajouter beaucoup de règles de surveillance, car ça ralentirait la bande passante du réseau d'entreprise.

Dans le cas où nous voulons customiser une règle, nous allons utiliser la fonction qui nous est proposée par le langage C dans ce dossier. Effectivement, si nous voulons ajouter des conditions pour une règle spécifique, nous pouvons sans autre.

Par exemple, nous pouvons ajouter, en langage C, une itération dans une règle de détection du contenu « POST », pour alerter l'utilisateur seulement après plusieurs détections du contenu « POST » dans le paquet.

3.6.1.2 Règles preproc_rules

En ce qui concerne les « preproc_rules », il s'agit des règles du préprocesseur qui permettent de désactiver ou bien d'activer les événements qui se déroulent dans le préprocesseur. On peut, avec ces règles, également spécifier le type de règle ou bien l'action d'un événement du décodeur ou bien du préprocesseur.

Les preproc_rules s'occupent du pré-traitement des paquets. Ils vont les assembler dans le cas où les paquets sont fragmentés. C'est utile pour avoir une meilleure détection par la suite.

Nous n'allons pas non plus gérer ce qu'il se passe dans le préprocesseur car dans notre cas de PME, nous n'en avons pas besoin.

3.7 Synthèse

Avec ces diverses configurations, on va donc pouvoir configurer une solution de surveillance de réseau avec Snort. Dans ce chapitre, plusieurs points essentiels ont été abordé, c'est-à-dire :

- Comment fonctionne Snort
- Qu'est-ce qu'une règle Snort
- Quelles sont les options à disposition pour une règle Snort
- Comment réagit Snort face à la mise-à-jour

Ce chapitre représente la partie théorique qui est importante pour poser les bases des composants que nous observons.

Nous allons maintenant voir ce qu'il faut faire pour installer et configurer ces composants, l'apport personnel pour la configuration d'une solution pour une PME.

4. Contribution personnelle

Dans ce chapitre, il sera question de la contribution personnelle pour ce travail. Le but étant de fournir une solution à une PME, qui permettrait de surveiller le trafic de paquets dans le réseau.

Donc, ma solution contient les thèmes suivants :

- Configuration de Snort
- Exemple d'écriture des règles spécifiques de surveillance
- Automatisation de la gestion de mise à jour des règles Snort.
- Lancement de Snort

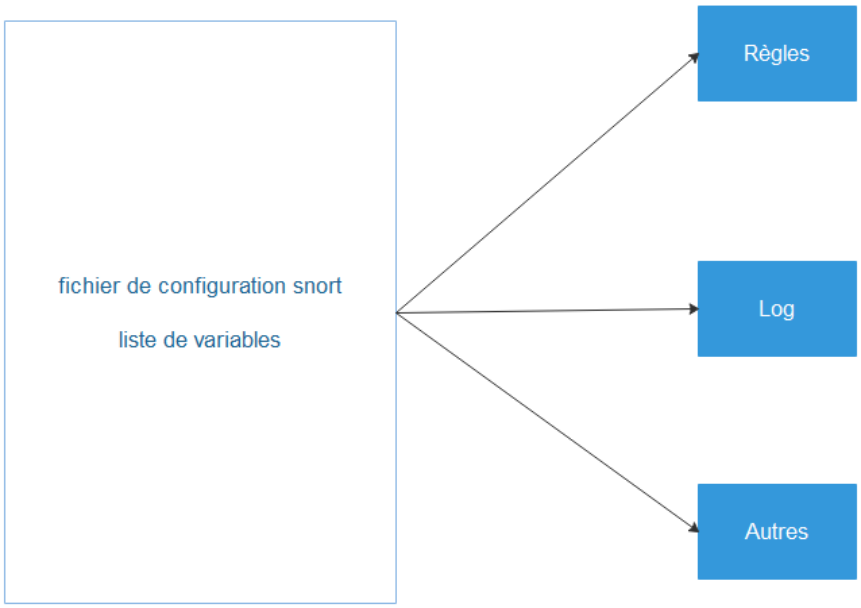
4.1 Configuration de Snort

Nous allons voir plus précisément le fichier « snort.conf » et nos choix de modifications et/ou d'ajout pour avoir une configuration qui nous permet de surveiller le réseau dans des conditions optimales.

Il est nécessaire de prendre le temps de comprendre la configuration de Snort, qui est un facteur essentiel. Il permet d'indiquer le réseau spécifique à surveiller, les chemins d'accès aux règles, aux logs et pleins d'autres indications.

Voilà comment visualiser la configuration de Snort par rapport à ses autres composants :

Figure 16 : Représentation de la configuration Snort



(Ilir Kadriu)

On peut donc voir sur la figure 16, l'importance du fichier de configuration de Snort qui fait le lien entre tous les autres composants de Snort. Nous allons devoir prendre des décisions, et faire des choix par rapport à notre PME et à notre volonté d'automatisme.

4.1.1 Réseau d'entreprise à surveiller HOME_NET

Premièrement, il faut définir un réseau à surveiller. Nous allons lui attribuer la variable HOME_NET. Cette première étape est très importante, car à la place d'écrire à chaque fois les règles avec l'adresse IP de chaque machine, nous allons plutôt définir un réseau en variable globale. Dans le fichier de configuration, on peut trouver cette variable dans l'étape 1 :

Figure 17 : variable HOME_NET

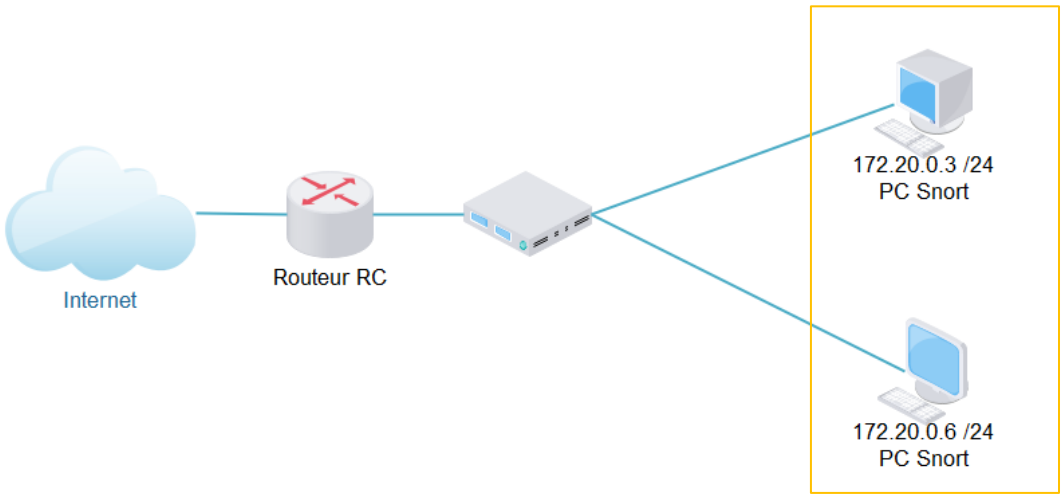
```
# Setup the network addresses you are protecting
ipvar HOME_NET 172.20.0.0/24
```

(Ilir Kadriu)

Dans la figure 17**,** on peut voir qu'il est important de spécifier le réseau alloué à l'entreprise. Par exemple : 172.20.0.0/24 qui est le réseau à surveiller.

On utilisera donc cette variable pour configurer le réseau tel que ci-dessous :

Figure 18 : Utilisation de la variable HOME_NET



(Ilir Kadriu)

On peut voir, sur la figure 18, que nous allons donc englober les périphériques qui se trouvent dans le réseau HOME_NET. Pour que le système Snort puisse être

maintenable, nous faisons donc appel à la variable HOME_NET pour ne pas devoir indiquer manuellement chaque adresses IP.

4.1.2 Réseau externe à surveiller EXTERNAL_NET

La variable EXTERNAL_NET définit la source, donc la provenance, du flux de paquets qui vient de l'extérieur du réseau de l'entreprise.

Figure 19 : Utilisation de la variable EXTERNAL_NET

```
# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
```

(Ilir Kadriu)

Donc dans la figure 19 nous pouvons constater qu'il est indiqué « **any** » c'est-à-dire que nous allons surveiller la provenance de n'importe quel émetteur. Nous pourrions très bien indiquer un émetteur particulier à surveiller. Le choix de laisser « **any »** apporte plus de sûreté car il permet de ne pas devoir adapter chaque fois la variable si l'adresse de l'émetteur change.

On pourrait aussi avoir un DMZ_NET avec une adresse pour spécifier le réseau DMZ à surveiller.

4.1.3 Les variables générales

Il y a d'autres variables comme le SSH_SERVER, HTTP_SERVER, ou autres que nous pouvons ajuster à notre entreprise, pour avoir une protection optimale, mais logiquement, ce serait dans une grande entreprise ou il y aurait plus de réseaux différents, donc il faudrait spécifier plusieurs variables pour chaque équipement. Dans une PME, on possède quelques adresses à surveiller, on peut tout simplement englober dans la variable HOME_NET toujours de manière logique, et laisser la variable EXTERNAL_NET par défaut avec la valeur **any** pour que l'on ait une utilisation optimale de nos règles.

4.1.4 Liste des ports à surveiller

On peut aussi ajuster notre surveillance à certains ports. Il est utile cette fois ci de les surveiller certains ports et pas tous, car par exemple, pour les règles qui concernent un serveur « http » par exemple, nous n'aurons pas besoin du port 22 qui est le port du protocole « SSH ».

Figure 20 : Variable pour les ports

```
# List of ports you run web servers on
portvar HTTP_PORTS [36,80,81,82,83,84,85,86,87,88,89,90,311]
```

(Ilir Kadriu)

Sur la figure 20, on peut donc observer un exemple de configuration de port dans le fichier « snort.conf ». Les crochets carrés nous indiquent une liste qui représente les ports utilisés par le protocole « http ».

Rappelons-nous, sur la figure 13, nous pouvions voir un ensemble de règle Snort pour le navigateur et si nous regardons de près, nous pouvons constater que la variable HTTP_PORTS est utilisée. Ce qui reste logique car on traiter des règles par rapport au navigateur.

4.1.5 Chemin d'accès aux règles Snort

On arrive à une étape très importante, celle de diriger le fichier de configuration vers les dossiers de règles Snort. Nous allons faire un choix primordial pour le partage de fichiers dans ce sous-chapitre. Premièrement, voyons à quoi ressemble l'attribution des chemins de règles, dans ce fichier de configuration.

Figure 21 : Chemin d'accès aux règles

```
# Path to your rules files (this can be a relative path)
# Note for Windows users:  You are advised to make this an absolute path,
# such as:  c:\snort\rules
var RULE_PATH C:\Snort\rules
var SO_RULE_PATH C:\Snort\so_rules
var PREPROC_RULE_PATH C:\Snort\preproc_rules
```

(Ilir Kadriu)

Donc ici, dans la figure 21, nous pouvons indiquer les chemins de <RULE_PATH>, <SO_RULE_PATH> et <PREPROC_RULE_PATH>. Nous allons faire le choix d'indiquer le chemin vers un dossier partagé de règles pour que nous puissions utiliser nos règles d'un dossier en commun avec tous les hôtes du réseau.

Pour ce faire, nous allons utiliser Samba sous Linux.

4.1.6 Partage de règles avec Samba

Avant d'entamer le sujet du partage de règles, il est important de souligner le fait que dans notre cas : pour sécuriser le trafic d'une PME, nous voulons des règles centralisées parce qu'on pourrait avoir une meilleure gestion des règles.

Effectivement, les règles Snort peuvent être utilisées par plusieurs hôtes Snort sur le réseau, c'est pourquoi il est important d'avoir un seul fichier de règles. Il est aussi important de souligner que les règles Snort ne sont pas les mêmes pour l'OS Windows

et l'OS Linux, c'est pourquoi il faut avoir un dossier centralisé de règles que nous diviserons en un dossier de règles Linux et un dossier de règles Windows.

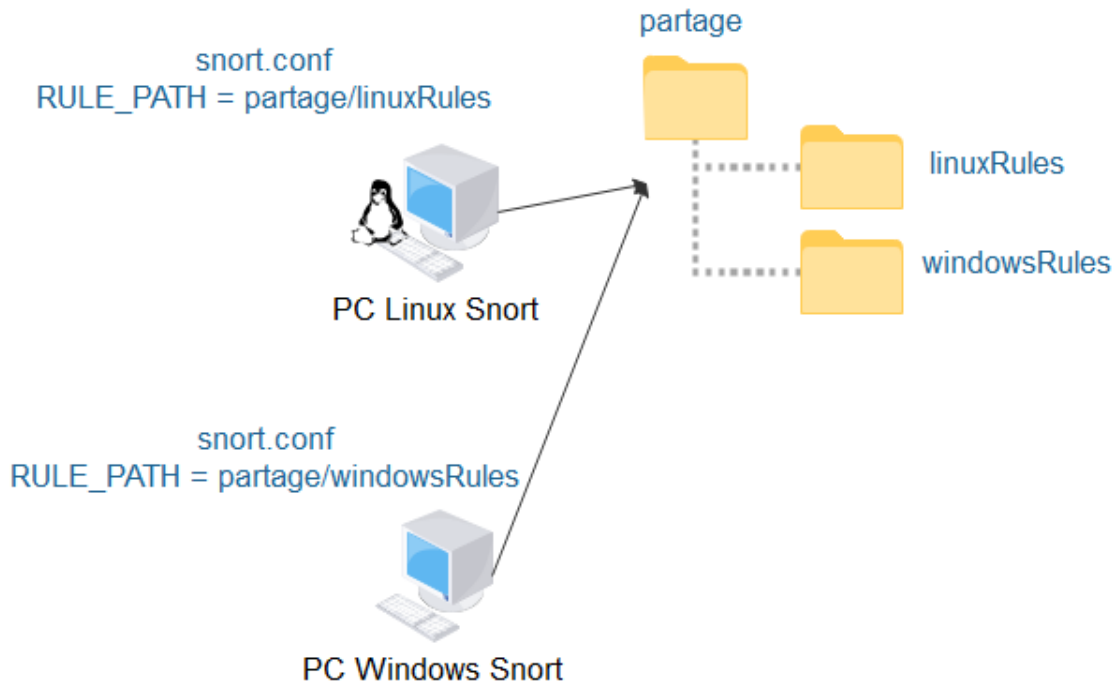
Pour mettre en place ce partage de dossier entre hôtes du réseau, nous allons utiliser **Samba :**

- Samba est un logiciel que l'on utilise pour Linux et qui va permettre l'accessibilité et le partage entre utilisateurs Windows et utilisateur Linux.

Avec Samba, nous allons donc pouvoir mettre en place un server Samba qui va donner un accès de lecture au dossier de règles Snort pour les utilisateurs du système.

Grâce à Samba, nous restons donc dans l'optique d'avoir un dossier de règles partagé entre hôtes Snort du système, même si ces derniers fonctionnent sous différents OS.

Figure 22 : Partage de règles avec Samba



(Ilir Kadriu)

Sur la figure 22, nous pouvons donc voir que le fichier de configuration amène à un dossier centralisé où se trouvent les règles Linux, et les règles Windows. C'est une question de maintenance lors de la mise-à-jour des règles, c'est pourquoi notre choix s'est orienté sur Samba qui permet ce partage de règles.

Cette configuration à deux dossiers n'est nécessaire que si les deux OS sont présents sur le réseau et nécessitent d'être surveillés.

Si la PME ne possède que des hôtes Linux, nous pouvons simplement mettre un dossier de règles Linux, s'l n'y a que des hôtes Windows, nous mettrons seulement un dossier de règles Windows, sinon nous intégrerons les deux dossiers.

Par défaut, notre choix se porte sur la création de deux dossiers de règles pour les deux OS, car pour la suite, si l'entreprise désire intégrer l'un ou l'autre des OS, il faut que la structure soit déjà existante. C'est un choix de bonne pratique.

4.1.6.1 Mise en place de Samba

Après avoir installé Samba sur un serveur Linux, j'ai créé un dossier partagé qui se nomme « rules » et intégré des utilisateurs pour la gestion des accès dans le fichier « etc/samba/smb.conf » :

Figure 23 : Configuration de Samba

```
[rules]
path = /home/ilir-kad/rules
valid users = ilir
writable = yes
read only = no
```

(Ilir Kadriu)

On peut voir dans la figure 23 que j'ai autorisé l'utilisateur qui s'authentifie en tant que « ilir » à accéder à mon dossier « rules ». C'est donc dans ce fichier que nous pouvons intégrer les utilisateurs du réseau pour qu'ils aient accès au dossier de règles.

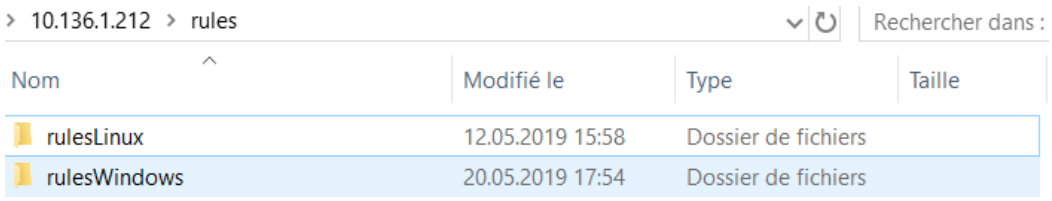
4.1.6.2 Test de Samba

J'ai accédé, sur le serveur Windows, d'accéder au partage en spécifiant ce chemin dans l'explorateur de fichier :

- \\10.136.1.212\rules

On obtient donc le résultat suivant :

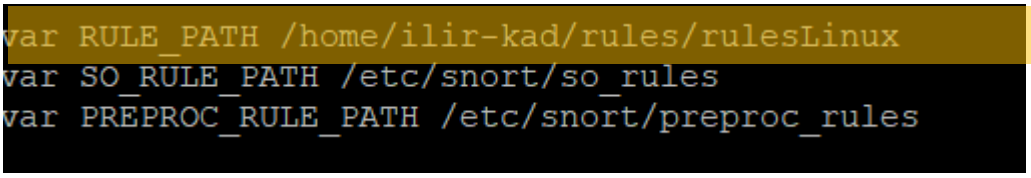
Figure 24 : Test de Samba



(Ilir Kadriu)

Selon la figure 24, le partage a bien fonctionné. Il suffit maintenant tout simplement de remplacer la variable RULE_PATH par le chemin indiqué ci-dessus selon l'OS que nous utilisons :

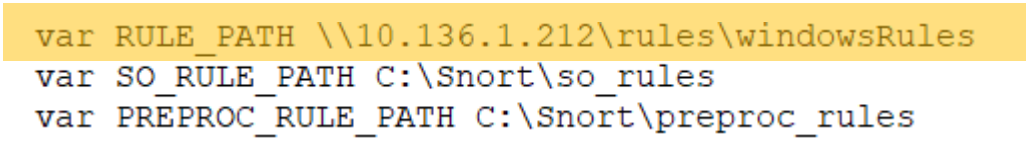
Figure 25 : Redirection de Snort Linux vers Samba



(Ilir Kadriu)

Voici, dans la figure 25, ce que nous allons retrouver au niveau du fichier snort.conf. Il est normal qu'on utilise directement le path du dossier Samba qui est configuré sur Linux même. Voyons maintenant pour Windows :

Figure 26 : Redirection de Snort Windows vers Samba



(Ilir Kadriu)

Dans la figure 26, nous retrouvons le lien vers le path Samba pour les utilisateurs Windows. Il faut souligner le fait que nous indiquons seulement le dossier « rules » et pas « so_rules » ou « preproc_rules », car comme indiqué précédemment, nous n'allons pas avoir besoin d'utiliser ces deux dossiers de règles là pour notre cas.

Il est utile de noter que Snort fonctionne aussi pour l'OS d'Apple, mais on n'entrera pas dans le détail dans ce travail, nous ne nous intéressons qu'à Windows et Linux.

Nous avons donc défini avec Samba, comment partager un dossier de règles, et comment centraliser toutes nos règles pour pouvoir être optimal dans l'utilisation des règles Snort.

4.1.7 Chemin d'accès aux Logs

Snort étant un logiciel fonctionnant avec des logs comme indiqué au chapitre trois, il faut aussi, toujours dans ce fichier de configuration, indiquer le chemin d'accès aux logs.

Ici aussi, nous pourrions faire le choix de mettre les logs dans un dossier centralisé, mais je me suis rendu compte de plusieurs choses concernant cette idée qui m'ont fait renoncer à cette idée :

- Les logs sont propres à chaque hôte du réseau

- Il y a beaucoup de logs générés, donc le dossier serait inondé de logs, par conséquent il serait très peu lisible
- Si nous voulions retrouver un log dans ce dossier pour un événement particulier, on perdrait beaucoup de temps et toutes les informations seraient emmêlées.

C'est pourquoi mon choix s'est dirigé vers la direction du dossier local sur chaque hôte C:/Snort/log :

Figure 27 : Redirection de log dans la configuration Snort

```
#
config logdir: C:\Snort\log
```

(Ilir Kadriu)

Nous observons, sur la figure 27, que les logs seront redirigés vers le dossier C:\Snort\log. C'est ici que nous pourrons donc retrouver les événements comme les alertes à examiner, ou bien les logs.

4.1.8 Inclure les règles spécifiquement

Nous allons maintenant passer à l'inclusion de règles dans le fichier de configuration de Snort. Dans cette partie de la configuration, nous avons le choix ou non d'inclure les fichiers de règles que nous souhaitons lors du lancement de Snort, et c'est un choix qui peut dépendre de la PME. Si la PME n'utilise pas le browser Firefox par exemple, il sera inutile de le prendre en considération dans le fichier de configuration et nous ajouterons simplement un « # » devant le fichier pour qu'il soit en commentaire et donc qu'il ne soit pas pris en compte.

Voici à quoi ressemble cette partie de la configuration :

Figure 28 : Inclusion de chaque règle

```
##################################################
# Step #7: Customize your rule set
# For more information, see Snort Manual, Writing Snort Rules
#
# NOTE: All categories are enabled in this conf file
##################################################

# site specific rules
include $RULE_PATH\local.rules

include $RULE_PATH\app-detect.rules
include $RULE_PATH\attack-responses.rules
include $RULE_PATH\backdoor.rules
include $RULE_PATH\bad-traffic.rules
include $RULE_PATH\blacklist.rules
include $RULE_PATH\botnet-cnc.rules
include $RULE_PATH\browser-chrome.rules
#include $RULE_PATH\browser-firefox.rules
```

(Ilir Kadriu)

Sur la figure 28, nous observons une liste non-exhaustive contenant les fichiers que nous ajouterons à notre RULES_PATH. On peut clairement voir deux indications très importantes :

- Le path acheminant à « browser-firefox.rules » est en commentaire car je simule que dans la PME, ils n'utilisent pas le navigateur Firefox.
- Le fichier local.rules qui indique les règles spécifiques à notre hôte.

Voici les modifications que j'ai effectué dans mon fichier « snort.conf » et les choix que j'ai pris pour adapter la configuration à une PME.

4.1.9 Lancement de Snort pour valider ma configuration

Pour tester si notre configuration a bien fonctionné, il suffit de lancer cette commande :

Figure 29 : Commande de lancement Snort

```
adminlx@Snort-Ilirkadriu:/etc/snort$ sudo snort -A console -c /etc/snort/snort.conf
```

(Ilir Kadriu)

Et nous arrivons donc à l'exécution de Snort qui attend qu'un événement se prodiuse :

Figure 30 : Snort running

```
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
Commencing packet processing (pid=3023)
```

(Ilir Kadriu)

Nous pouvons donc constater sur les figures 29 et 30, que Snort s'est bien lancé, et il n'y a pas eu de problèmes dans le fichier de configuration.

Dans cet exemple, nous avons utilisé la commande suivante pour lancer Snort :

- Sudo snort -A console -c /etc/snort/snort.conf

Cette commande stipule que nous lançons snort via le fichier de configuration, et que nous affichons les alertes de nos règles dans la console Linux.

4.2 Écriture des règles spécifiques de surveillance

Pour une entreprise, il est important d'avoir des règles spécifiques. En effet, chaque entreprise fonctionne différemment, c'est pourquoi il est primordial que Snort possède la fonction d'écrire ses propres règles qui s'appliquent à la politique de sécurité spécifique.

Comme vu au chapitre trois, nous avons défini les options de règles. Nous allons maintenant voir quelques exemples de règles que nous pouvons intégrer à Snort pour qu'il surveille le réseau de notre PME selon ses objectifs de sécurités spécifiques.

Par exemple, si dans notre PME, le patron s'absente souvent en voyage d'affaire, et qu'il veut accéder au réseau d'entreprise via SSH, nous devons spécifier des règles qui permettent de surveiller les ports SSH car on sait que ce protocole sera souvent utilisé.

4.2.1 Exemple de règles spécifiques à l'entreprise

Dans le cadre de notre surveillance, nous voulons préciser une règle de surveillance. Dans notre cas, nous voulons que toute tentative de connexion SSH soit détectée par Snort et qu'une alerte soit générée. Nous allons donc écrire cette règle dans le fichier « local.rules » car c'est ici que nous allons spécifier nos règles personnelles :

Figure 31 : Règle d'intrusion SSH

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"Connexion Attempt"; sid:10055;)
```

(Ilir Kadriu)

Sur la figure 31, nous observons donc que Snort va analyser tous les paquets TCP provenant de l'extérieur avec la variable <EXTERNAL_NET> et qui se dirigent vers notre réseau d'entreprise <HOME_NET> par le port 22. Nous avons donc optimisé la règle avec le port SSH et le protocole utilisé par SSH.

Voici ce que résulte de cette règle en lançant Snort et en tentant une connexion depuis l'adresse 10.136.153.169 :

Figure 32 : Résultat de surveillance SSH

```
06/02-23:48:00.117889  [**] [1:10055:0] Connexion Attempt [**] [Priority: 0] {TCP} 10.
136.153.196:51166 -> 10.136.1.212:22
06/02-23:48:01.076652  [**] [1:10055:0] Connexion Attempt [**] [Priority: 0] {TCP} 10.
136.153.196:51166 -> 10.136.1.212:22
06/02-23:48:01.124948  [**] [1:10055:0] Connexion Attempt [**] [Priority: 0] {TCP} 10.
136.153.196:51166 -> 10.136.1.212:22
06/02-23:48:02.077516  [**] [1:10055:0] Connexion Attempt [**] [Priority: 0] {TCP} 10.
136.153.196:51166 -> 10.136.1.212:22
06/02-23:48:02.118765  [**] [1:10055:0] Connexion Attempt [**] [Priority: 0] {TCP} 10.
136.153.196:51166 -> 10.136.1.212:22
```

(Ilir Kadriu)

Nous pouvons donc voir sur la figure 32 qu'il y a eu des tentatives de connexion avec le message que nous avons indiqué au préalable : « Connexion Attempt », et l'adresse IP de l'EXTERNAL_NET 10.136.153.196 qui se dirige vers l'IP de l'HOME_NET.

Cette règle est un exemple simple de ce qui pourrait se trouver sur le fichier « local.rules ».

Nous pourrions aussi avoir un exemple de règle dynamique pour la connexion SSH :

Figure 33 : Règle dynamique pour la détection de connexion SSH

```
activate tcp $EXTERNAL_NET any -> $HOME_NET 21:22 (msg:"Connexion attempt"; sid:100005; activates:1;)
dynamic tcp $HOME_NET any -> any any (activated_by:1; count:50;sid:100006;)
```

(Ilir Kadriu)

Sur la figure 33, nous constatons que la règle activate va s'activer seulement après la détection du dynamique :

Figure 34 : Résultat de règle dynamique

(Ilir Kadriu)

Sur la figure 34, l'utilisateur est averti par la règle dynamique. Il reçoit donc des alertes indiquant qu'il y a eu plusieurs tentatives de connexion.

```
WARNING: an activation rule with no dynamic rules matched.
06/03-00:09:28.071672  [**] [1:100005:0] Connexion attempt [**] [Priority: 0] {TCP} 10.136.153.196:5116
6 -> 10.136.1.212:22
WARNING: an activation rule with no dynamic rules matched.
06/03-00:09:28.071776  [**] [1:100005:0] Connexion attempt [**] [Priority: 0] {TCP} 10.136.153.196:5116
6 -> 10.136.1.212:22
```

Lorsque nous arrêtons l'exécution de Snort, des statistiques nous indiquent les événements ayant eu lieu :

Figure 35 : Statistique post exécution

```
Action Stats:
     Alerts:            201 ( 19.706%)
     Logged:              0 (  0.000%)
     Passed:              0 (  0.000%)
Limits:
      Match:              0
      Queue:              0
        Log:              0
      Event:              0
      Alert:              0
```

(Ilir Kadriu)

Nous observons sur la figure 35 affichant les statistiques de l'exécution de Snort, qu'il y a eu 201 alertes déclenchées par rapport aux tentatives de connexion SSH.

4.2.2 Constat sur les règles

L'écriture de règles Snort dans le fichier de règles locales, doit correspondre à ce que nous voulons surveiller de manière plus spécifique à un équipement réseau dans notre PME. Par exemple, s'il faut surveiller un périphérique qui n'est pas intégré dans les règles de bases de Snort, nous écrirons une règle spécifique dépendant de ce que nous voulons comme surveillance. Grâce au chapitre trois et à l'explication de règles et ses composants (adressage IP, ports, options), nous pourrions donc écrire les règles que nous souhaitons pour protéger notre PME.

Nous avons, tout au long de ce travail, discuté de la mise-à-jour de règles avec de nouvelles signatures découvertes quotidiennement. Il faut donc maintenant s'intéresser au moyen que l'on a pour mettre en place cette mise-à-jour de manière automatique avec le composant PulledPork.

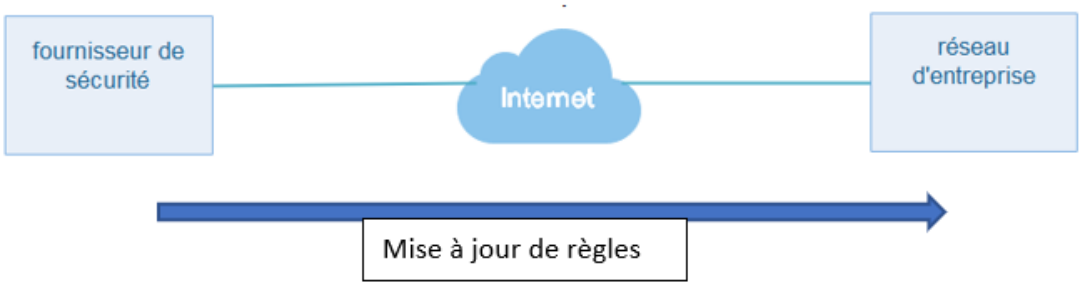
4.3 Automatisation de mise à jour de règle avec PulledPork

PulledPork est un module qui va nous permettre de rester dans l'optique d'avoir un fournisseur de service qui s'occupe de mettre à jour les règles depuis l'extérieur de notre réseau d'entreprise.

4.3.1 Visualiser la mise-à-jour de PulledPork

Voici comment représenter par un schéma, la configuration que nous souhaitons réaliser avec PulledPork :

Figure 36 : Objectif de mise à jour



(Ilir Kadriu)

Nous observons donc, sur la figure 36, qu'avec PulledPork, nous allons pouvoir importer automatiquement des nouvelles règles Snort qui apparaissent quotidiennement.

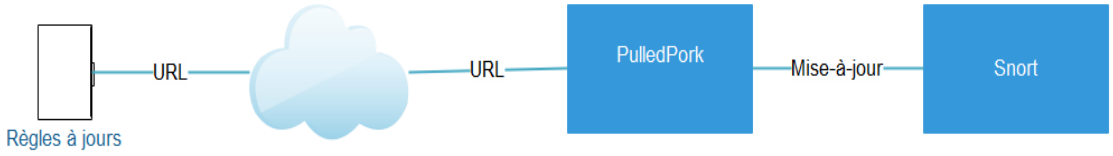
4.3.2 Comment fonctionne PulledPork

PulledPork se définit à travers un fichier de configuration qui va indiquer les liens URL qui amène à des fichiers téléchargeables, contenant des règles à jour, et remplaçant les anciens fichiers de règles par ces nouvelles règles.

Il est important de noter que seul le fichier local.rules ne sera pas impacté et c'est celui-ci que le fournisseur va pouvoir personnaliser en fonction de la PME et ses besoins.

Voici comment fonctionne PulledPork :

Figure 37 : Fonctionnement de PulledPork



(Ilir Kadriu)

Nous remarquons, à travers la figure 37, que PulledPork va mettre à jour les règles, via des URL qui amènent à Snort.org.

Nous allons récupérer les règles Snort, pour que Snort soit mis à jour quotidiennement via les nouvelles attaques répertoriées à travers les règles.

4.3.3 Fichier de configuration de PulledPork

Après avoir installé PulledPork sur Linux, il faut modifier son fichier de configuration. Il faut premièrement indiquer l'URL de snort.org avec le dossier de règles que nous voulons importer quotidiennement, comme ci-dessous :

Figure 38 : Route URL de PulledPork

```
# i.e. url|tarball|123456789,
rule_url=https://www.snort.org/reg-rules/|snortrules-snapshot-29130.tar.gz|18bcad5
d9a194bf39582caf8b669017436d8d40b
```

(Ilir Kadriu)

Sur la figure 38, nous voyons le path amenant à « snortrules-snapshot-29130 » qui est le path officiel de snort.org pour quérir les nouvelles règles qui vont mettre à jour les anciennes règles.

Lorsque des nouvelles signatures seront enregistrées par Snort sur leur site, nous allons donc y avoir accès et être à jour par rapport aux nouvelles attaques qui apparaissent.

Nous apercevons un code après le dossier « tar » de règles Snort : C'est le Oinkcode de Snort. Lorsque l'on veut télécharger des règles Snort, il faut avoir un Oinkcode pour que l'on puisse s'identifier à Snort et pour que Snort reconnaisse notre compte que l'on a créé au préalable sur snort.org. C'est pourquoi il faut indiquer un oinkcode et que l'on ne peut pas télécharger automatiquement ce que l'on souhaite sur n'importe quel autre serveur.

Il est important de rappeler que seul le fichier « local.rules » ne sera pas modifié, et nous verrons dans le prochain sous-chapitre, le problème de la mise à jour de ce fichier spécifique à chaque hôte.

Ensuite, il faut évidemment indiquer à la configuration de PulledPork l'endroit où il doit mettre ses nouvelles règles en écrasant les anciennes, donc dans la suite de la configuration, on doit modifier le rule_path comme suit :

Figure 39 : Lien de PulledPork à Snort

```
rule_path=/etc/snort/rules/snort.rules
```

(Ilir Kadriu)

Nous remarquons sur la figure 39 que le path achemine à notre dossier de règles de base, cependant, nous voulons l'acheminer directement à notre dossier partagé Samba, donc nous allons modifier cela comme suit :

Figure 40 : Lien de PulledPork à Samba

```
rule_path=/home/ilir-kad/rules/rulesLinux
```

(Ilir Kadriu)

Sur la figure 40, nous modifions le path pour qu'il se dirige vers notre dossier partagé par Samba. Évidemment, il faut aussi que nous indiquions où se trouvent d'autres composants Snort comme le fichier de configuration que nous avons traité dans le sous-chapitre précédant ou le path qui redirige au bin de Snort pour l'exécutable :

Figure 41 : Autre path à modifier sur PulledPork

```
snort_path=/usr/sbin/snort

# We need to know where your snort.conf file lives so that we can
# generate the stub files
config_path=/etc/snort/snort.conf
```

Sur la figure 41, nous modifions donc les derniers path de direction à Snort sur PulledPork.

4.3.4 Planification de mise à jour de règle

Nous pouvons planifier la mise à jour de règles avec PulledPork. Il suffit, pour cela, d'intégrer la commande suivante :

- #sudo crontab -e
- #01 04 *** /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf -l

Cette commande planifie le lancement de PulledPork quotidiennement à 04 :01 AM.

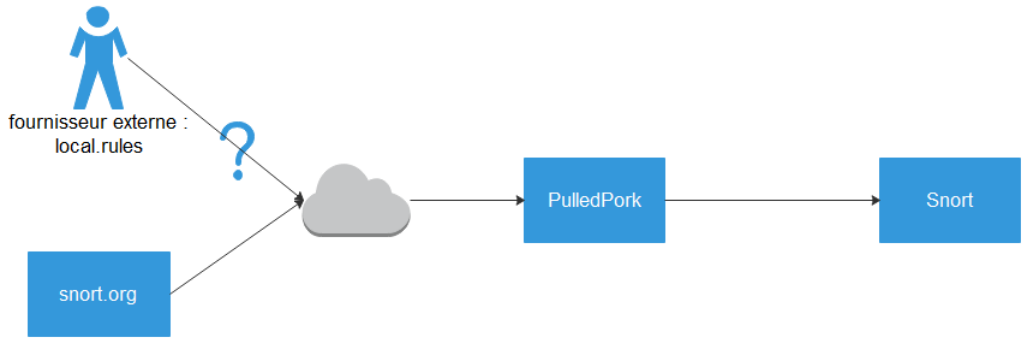
4.3.5 Problème du fichier de configuration local

PulledPork nous offre la possibilité d'avoir une mise-à-jour des signatures Snort quotidiennement, cependant il persiste un problème :

- Dans notre cas, le fournisseur de services ne peut pas utiliser PulledPork pour mettre à jour ses règles personnalisées car PulledPork ne fonctionne qu'avec snort.org

Donc j'arrive à ce constat-là :

Figure 42 : Problème de mise à jour local



(Ilir Kadriu)

Nous constatons, sur la figure 42, que nous n'avons pas accès automatiquement au fichier local.rules que le fournisseur aurait mis à jour régulièrement. Le problème étant donc que PulledPork peut seulement prendre ses règles du site officiel Snort.

4.3.5.1 Ce qu'on pourrait faire par la suite

Une solution serait d'avoir un serveur du côté du fournisseur, qui fournirait un lien de téléchargement de règles locales, pour que, de l'autre côté, le client puisse,

quotidiennement, récupérer ces règles via une URL sécurisée et de confiance, pour mettre à jour les règles locales.

Nous pourrions, par la suite de ce travail, mettre en place ce système en utilisant, du côté du client, la fonction **wget**, avec la fonction **crontab** que nous avons déjà abordé.

Wget est un client http qui permet de recueillir le contenu de ce qu'offre un serveur Web. Donc grâce à cette fonction, nous allons pouvoir, du côté du fournisseur de sécurité, écrire les règles locales et spécifiques à notre client, et du côté du client, nous allons simplement mettre en place la récupération de ces règles locales avec wget, puis indiquer un planning avec crontab pour que l'on puisse instaurer un automatisme régulier de mise-à-jour de nos règles locales.

4.4 Synthèse

Dans ce chapitre, nous avons donc vu ma contribution personnelle pour ce travail, et les choix effectués pour atteindre notre but initial. Nous avons donc abordé :

- Samba pour le partage de dossier de règles
- L'écriture de règles locales
- PulledPork pour la mise à jour de règles quotidiennement
- Mes choix de configuration de Snort et PulledPork

5. Conclusion

Lors de ce travail de Bachelor, j'ai pu me familiariser avec l'IDS Snort. J'ai pu remarquer l'importance de la configuration de ce dernier, et j'ai observé qu'il y a des fonctionnalités que l'on peut faire notamment grâce à PulledPork, mais ce Framework ne permet pas non plus de faire tout ce que l'on a souhaité lorsque nous avons fixé les objectifs de ce travail. Snort est un NIDS qui permet la surveillance de notre réseau, et que l'on configure en ajoutant des règles pour optimiser la surveillance du réseau de PME.

Tout au long de ce travail, nous avons approfondi 3 problématiques :

- Pourquoi intégrer la surveillance du réseau dans une PME.
- La centralisation des règles pour une meilleure gestion de surveillance.
- La mise-à-jour des règles via un fournisseur externe.

Concernant l'importance de la surveillance de réseau d'entreprise, nous avons pu voir qu'une PME possédant un réseau d'entreprise, doit utiliser des outils de sécurité. Elle utilise déjà certainement certains outils comme un Firewall ou autre outil, mais la surveillance du réseau par un IDS est un moyen d'alerter l'entreprise qu'une attaque peut avoir lieu dans le cas où ses autres outils n'ont pas bloqué l'attaque.

La centralisation des règles a été abordé et nous avons fournis une solution avec Samba. C'est une question de gestion de sécurité, il est toujours plus simple d'avoir une bonne gestion si les fichiers sont centralisés.

Concernant la mise à jour, le Framework PulledPork nous a permis d'effectuer une partie seulement de la mise à jour, mais ce Framework ne nous a pas permis d'avoir une mise-à-jour des fichiers de règles locales, que le fournisseur de sécurité pourrait lui-même écrire.

Ce qu'il pourrait être fait pour continuer cette recherche, serait justement de voir une solution d'automatisation de mise à jour du fichier de règles locales comme nous l'avons abordé au chapitre quatre, avec une solution grâce à la fonction wget.

Ce travail m'a permis d'avoir une meilleure connaissance du réseau d'entreprise, de la sécurité et surtout de la surveillance du réseau d'entreprise pour une PME. Je ne connaissais pas le concept d'IDS, ou bien même Snort et c'est avec plaisir que j'ai appris à utiliser ce système. J'ai vaguement hésité sur mon utilisation de Snort en IPS ou IDS, et j'ai fini par utiliser la partie surveillance du réseau et non pas l'action qui serait immédiate.

L’apport personnel m’a donné l’occasion de ma familiariser avec l’IDS Snort, le Framework PulledPork, mais aussi Samba et les différentes fonctions de Linux.

Le sujet de la sécurité est très vaste et j’ai exploré seulement une partie de ce sujet. Le fait de ne pas m’être concentré sur d’autres outils de sécurité est un choix, car je pense que la surveillance et l’analyse sont des sujets très importants au vu du développement d’attaques et d’intrusions dans les réseaux d’entreprises.

Bibliographie

About Network IDS (NIDS) in AlienVault USM Appliance [en ligne]. [Consulté le 22 Mars 2019]. Disponible à l'adresse : https://www.alienvault.com/documentation/usm-appliance/ids-configuration/about-alienvault-nids.htm

Déployer un système de détection d'intrusion – ZDNet [en ligne]. [Consulté le 24 Mars 2019]. Disponible à l'adresse : https://www.zdnet.fr/actualites/deployer-un-systeme-de-detection-d-intrusion-2118189.htm

Difference between ACL on Router and Firewall | IP With Ease [en ligne]. [Consulté le 4 Avril 2019]. Disponible à l'adresse : https://ipwithease.com/difference-between-acl-on-router-and-firewall/

GitHub - shirkdog/pulledpork : Pulled Pork for Snort and Suricata rule management (from Google code) [en ligne]. [Consulté le 5 Mai 2019]. Disponible à l'adresse : https://github.com/shirkdog/pulledpork

IDS vs. IPS : What is the Difference? [en ligne]. [Consulté le 4 Avril 2019]. Disponible à l'adresse : https://www.varonis.com/blog/ids-vs-ips/

Install and Configure Samba | Ubuntu tutorials. [en ligne]. [Consulté le 28 Avril 2019]. Disponible à l'adresse : https://tutorials.ubuntu.com/tutorial/install-and-configure-samba

Intrusion detection system. *Wikipedia : l'encyclopédie libre* [en ligne].Dernière modification de la page le 18 mai 2019 à 08 :16. [Consulté le 18 Mars 2019]. Disponible à l'adresse : https://en.wikipedia.org/wiki/Intrusion_detection_system

IPTABLES VS FIREWALLD | Unixmen [en ligne]. [Consulté le 4 Avril 2019]. Disponible à l'adresse : https://www.unixmen.com/iptables-vs-firewalld/

La sécurité du réseau informatique en entreprise : c'est quoi ? pourquoi est-ce très important ? [en ligne]. [Consulté le 15 Mars 2019]. Disponible à l'adresse : http://www.socrateand.com/la-securite-du-reseau-informatique/

Les IDS par la pratique : Snort [en ligne]. [Consulté le 25 Mars 2019]. Disponible à l'adresse : http://www-igm.univ-mlv.fr/~dr/XPOSE2004/IDS/IDSSnort.html

Managed IDS | Swisscom – Managed Network Security (MSS-i) [en ligne]. [Consulté le 24 Mai 2019]. Disponible à l'adresse : http://mss-i.swisscom.ch/en/services/managed-ids/

Pare-feu (informatique). *Wikipedia : l'encyclopédie libre* [en ligne].Dernière modification de la page le 7 avril 2019 à 14 :35. [Consulté le 18 Mai 2019]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Pare-feu_(informatique)

SCOTT, Charlie, WOLFE, Paul and HAYES, Bert, 2004. *Snort for dummies*. Hoboken, NJ : Wiley Pub. --For dummies. ISBN 978-0-7645-6835-0.

TK5105.59. S388 2004

SNORT 1 : NIDS presentation, NIDS network topology - Computer Outlines Blog [en ligne]. [Consulté le 15 Mars 2019]. Disponible à l'adresse : http://computer-outlines.over-blog.com/article-snort-1-nids-presentation-nids-network-topology-122011437.html

Snort 2.9.9.x on Ubuntu – Part 5 : Installing PulledPork – Sublime Robots [en ligne]. [Consulté le 15 Mai 2019]. Disponible à l'adresse : http://sublimerobots.com/2017/01/snort-2-9-9-x-ubuntu-installing-pulledpork/

Snort (software). *Wikipedia : l'encyclopédie libre* [en ligne].Dernière modification de la page le 11 avril 2019 à 19 :46. [Consulté le 13 Mars 2019]. Disponible à l'adresse : https://en.wikipedia.org/wiki/Snort_(software)

Snort, un IDS Open-source. | SUPINFO, École Supérieure d'Informatique [en ligne]. [Consulté le 13 Mars 2019]. Disponible à l'adresse : https://www.supinfo.com/articles/single/8421-snort-ids-open-source

Systèmes de détection et de prévention d'intrusion - 1&1 IONOS [en ligne]. [Consulté le 13 Mars 2019]. Disponible à l'adresse : https://www.ionos.fr/digitalguide/serveur/securite/systemes-de-detection-et-de prevention-dintrusion/

The Beginner's Guide to iptables, the Linux Firewall [en ligne]. [Consulté le 22 Avril 2019]. Disponible à l'adresse : https://www.howtogeek.com/177621/the-beginners-guide-to-iptables-the-linux-firewall/

The Differences Between a Firewall and an Intrusion Detection System | Chron.com [en ligne]. [Consulté le 5 Avril 2019]. Disponible à l'adresse : https://smallbusiness.chron.com/differences-between-firewall-intrusion-detection-system-62856.html

Annexe 1 : Installation de Snort

L'installation de Snort est différente dépendant de l'OS que nous utilisons. Voici comment installer Snort sur :

- Linux
- Windows

Snort sur Linux

Pour installer Snort sur Linux, il faut d'abord préparer le serveur pour le déploiement, et donc, premièrement, nous devons nous enregistrer dans le root user et mettre à jour le système :

- apt-get update -y
- apt-get upgrade -y

Ensuite, il faut installer les dépendances requises sur le système :

- apt-get install openssl-server ethtool build-essential libpcap-dev libpcre3-dev libdumbnet-dev bison flex zlib1g-dev liblzma-dev openssl libssl-dev
- wget *https://www.snort.org/downloads/snort/daq/daq-2.0.6.tar.gz*

Par la suite, il faut extraire les fichiers téléchargés du DAQ :

- tar -zxvf daq-2.0.6.gz
- cd daq-2.0.6

Ensuite, il faut faire la compilation et l'installation du DAQ :

- ./configure && make && make install

On va maintenant télécharger, extraire et installer Snort :

- wget https://www.snort.org/downloads/snort/snort-2.9.8.3.tar.gz
- tar -xvzf snort-2.9.8.3.tar.gz
- cd snort-2.9.8.3
- ./configure –enable-sourcefire && make && make install

Ensuite il faut mettre à jour la librairie partagée :

- Ldconfig

Puis créer un lien symbolique pour le bin de Snort :

- Ln -s /usr/local/bin/snort /usr/sbin/snort

Ensuite, on peut vérifier que Snort fonctionne avec la commande

- Snort -V

Snort sur Windows

Pour télécharger et installer Snort sur Windows, il faut aller sur le site :

- https://www.snort.org

Les étapes d'installations sont clairement indiquées sur le site, il suffit donc de les suivre pour télécharger et installer Snort sur un hôte Windows.

Annexe 2 : Installation de Samba sur Linux

Pour utiliser Samba sur Linux, nous allons passer par 3 étapes :

- L'installation de Samba
- La configuration de Samba
- La configuration des comptes utilisateurs de Samba

Installation de Samba

- sudo apt update
- sudo apt install samba
- whereis samba

Configuration de Samba

Il faut maintenant créer et configurer le dossier que nous voulons partager :

- mkdir /home/<utilisateur>/sambashare

Ensuite nous allons configurer le fichier de configuration de samba pour ajouter le chemin du dossier partagé :

- Sudo nano /etc/samba/smb.conf

Au milieu du fichier ajoutons ce qui suit :

[sambashare]

comment = Samba on Ubuntu

path = /home/<utilisateur>/sambashare

read only = no

browsable = yes

Ensuite, il faut relancer le service Samba :

- sudo service smbd restart

Configuration des comptes utilisateurs de Samba

Il faut maintenant ajouter un mot de passe pour l'utilisateur :

- sudo smbpasswd -a <utilisateur>

On peut tester en entrant l'adresse IP du serveur Linux et en ajoutant le nom du dossier :

- \\<adresse IP>\sambashare

Annexe 3 : Installation de PulledPork sur Linux

Il faut d'abord installer les prérequis de PulledPork :

- sudo apt-get install -y libcrypt-ssleay-perl liblwp-useragent-determined-perl

Ensuite, il faut télécharger PulledPork :

- cd ~/snort_src
- wget https://github.com/shirkdog/pullepork/archive/master.tar.gz -O pullepork-master.tar.gz
- tar xzvf pulledpork-master.tar.gz
- cd pulledpork-master/
- sudo cp pulledpork.pl /usr/local/bin
- sudo chmod +x /usr/local/bin/pulledpork.pl
- sudo cp etc/*.conf /etc/snort

Tester l'installation en lançant PulledPork :

- /usr/local/bin/pulledpork.pl -V